Compositional and Effectful Recursive Specification Formats

Distributive Laws and the Semantics of Recursion

Von der

Carl-Friedrich-Gauß-Fakultät

der Technischen Universität Carolo-Wilhelmina zu Braunschweig

zur Erlangung des Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigte Dissertation

von

Daniel Schwencke

geboren am 17.01.1982

in Berlin-Spandau

Eingereicht am: 10.04.2013

Disputation am: 13.09.2013

1. Referent: Prof. Dr. Jiří Adámek

2. Referent: Prof. Dr. Lutz Schröder

2014

Abstract

Recursive specifications are a powerful tool mainly used in mathematics and computer science. In the last decade, it was observed that many of them form coalgebras for a functor [Rut00]. A category theoretic approach to these specifications—to (systems of) recursive equations [Mil05, AMV06b] and to recursive program schemes [GLM03, MM06]—was developed. Our thesis carries this line of research further in two different directions by combining it with distributive laws [Bec69, TP97, LPW00].

In the first part of the thesis, we consider distributive laws of algebra functors over coalgebra functors. As shown in Bartels' Ph.D. thesis [Bar04] (based on results by Turi and Plotkin [TP97]), these define algebraic operations on the final coalgebra. We present several formats of recursive specifications which make use of operations defined that way. These formats either specify elements of the final coalgebra or further operations on it. An overview of the former—formats of recursive equations—is given on pages 58 and 92; for the latter—formats of recursive program schemes—see page 124. Following the category theoretic approach to recursive specifications and working with completely iterative algebras [Mil05], we obtain compositionality results for our specification formats (this is summarized in Remarks 4.9, 4.15 and 4.70(2) for the formats of recursive equations, and in Remarks 5.5, 5.22(3) and 5.29 for the formats of recursive program schemes). We demonstrate our results on five important special cases and show e. g. in Section 4.2.3 that Milner's solution theorem [Mil89] for his calculus of communicating systems (CCS) follows from our work. We also obtain in Section 5.2.1 a proof that Rutten's behavioral differential equations for the specification of operations on streams can be used in a step-by-step manner, a technique which has been applied by Rutten in his stream calculus [Rut05a].

In the second part of the thesis, we consider distributive laws of algebra functors over certain monads. The monads model computational effects in the sense of Moggi's work [Mog91], e. g. nondeterminism, and the distributive laws extend algebraic operations to cope with computational effects. We first see—mainly as a consequence of Theorem 6.19—that for five concrete monads canonical distributive laws exist. Adapting Milius' completely iterative algebras [Mil05] to our setting, we introduce (Definitions 7.2 and 7.5) two notions of algebras with effects in which systems of recursive equations have unique solutions. For some concrete effects we give a characterization of these algebras: partial algebras with unique solutions are characterized in Theorem 7.47, nondeterministic algebras in Theorem 7.50 and composite algebras

in Theorem 7.53. Finally we introduce (Definitions 8.48 and 8.53) the notion of a recursive program scheme with effects for all of our concrete effects and prove that all the guarded schemes have a unique or at least a canonical uninterpreted solution in Theorems 8.50, 8.60 and 8.63. For nonempty nondeterministic schemes, we compare our category theoretic approach with the classical one by Arnold and Nivat [AN80], see Remarks 8.57 and 8.62.

Zusammenfassung

Rekursive Spezifikationen sind ein leistungsstarkes Werkzeug, das vor allem in der Mathematik und Informatik genutzt wird. Im Laufe der letzten etwa zehn Jahre wurde entdeckt, dass viele von ihnen Koalgebren für einen Funktor [Rut00] darstellen. Ein kategorientheoretischer Ansatz für solche Spezifikationen – (Systeme) rekursive(r) Gleichungen [Mil05, AMV06b] und rekursive Programmschemata [GLM03, MM06] – wurde entwickelt. Die vorliegende Arbeit erweitert diese Forschung in zwei verschiedene Richtungen, indem sie sie mit Distributivgesetzen [Bec69, TP97, LPW00] kombiniert.

Im ersten Teil der Arbeit werden Distributivgesetze eines Algebra-Funktors über einem Koalgebra-Funktor betrachtet. Bartels zeigte in seiner Dissertation [Bar04] (aufbauend auf Ergebnisse von Turi und Plotkin [TP97]), dass solche Distributivgesetze algebraische Operationen auf finalen Koalgebren definieren. In der vorliegenden Arbeit werden verschiedene Formate rekursiver Spezifikationen präsentiert, in denen Operationen verwendet werden, die auf diese Weise definiert wurden. Diese Formate spezifizieren entweder Elemente der finalen Koalgebra oder weitere Operationen auf der finalen Koalgebra. Eine Übersicht über erstere – Formate rekursiver Gleichungen – findet sich auf den Seiten 58 und 92; für letztere – Formate rekursiver Programmschemata – siehe Seite 124. Indem der kategorientheoretische Ansatz für rekursive Spezifikationen verwendet und mit vollständig iterativen Algebren [Mil05] gearbeitet wird, kann gezeigt werden, dass die Spezifikationsformate die Eigenschaft der Kompositionalität besitzen (die Bemerkungen 4.9, 4.15 und 4.70(2) fassen dies für die Formate rekursiver Gleichungen zusammen, die Bemerkungen 5.5, 5.22(3) und 5.29 für die Formate rekursiver Programmschemata). Die Ergebnisse werden anhand von fünf wichtigen Spezialfällen illustriert. Beispielsweise wird in Abschnitt 4.2.3 gezeigt, dass Milners Lösungssatz [Mil89] für sein Prozess-Kalkül CCS aus den Ergebnissen folgt. Ein anderes Beispiel findet sich in Abschnitt 5.2.1, in dem bewiesen wird, dass Ruttens behavioral differential equations auch schrittweise für die Spezifikation von Operationen auf Streams verwendet werden können, ein Vorgehen, das von Rutten im Rahmen seines Stream-Kalküls [Rut05a] angewendet wurde.

Im zweiten Teil der Arbeit werden Distributivgesetze von Algebra-Funktoren über verschiedenen Monaden betrachtet. Die Monaden modellieren

Berechnungseffekte im Sinne der Arbeiten von Moggi [Mog91], z. B. Nichtdeterminismus, und die Distributivgesetze erweitern algebraische Operationen so, dass sie Berechnungseffekte verarbeiten können. Zunächst wird festgestellt, dass – hauptsächlich aufgrund von Satz 6.19 – für fünf konkrete Monaden kanonische Distributivgesetze existieren. Als Zweites werden Milius' vollständig iterative Algebren [Mil05] für die Arbeit mit Berechnungseffekten adaptiert und zwei Arten von Algebren mit Berechnungseffekten eingeführt (Definitionen 7.2 und 7.5), in denen Systeme rekursiver Gleichungen eindeutige Lösungen haben. Für einige konkrete Effekte werden diese Algebren charakterisiert: partielle Algebren mit eindeutigen Lösungen werden in Satz 7.47 charakterisiert, nichtdeterministische Algebren in Satz 7.50 und zusammengesetzte Algebren in Satz 7.53. Schließlich wird der Begriff eines rekursiven Programmschemas mit Effekten für alle fünf betrachteten konkreten Effekte eingeführt (Definitionen 8.48 und 8.53) und bewiesen, dass all diese Schemata, falls sie die Guardedness-Eigenschaft haben, eine eindeutige oder zumindest eine kanonische uninterpretierte Lösung besitzen, siehe die Sätze 8.50, 8.60 und 8.63. Für nichtleere nichtdeterministische Schemata wird der gewählte kategorientheoretische Ansatz mit dem klassischen Ansatz von Arnold und Nivat [AN80] verglichen (Bemerkungen 8.57 und 8.62).

Contents

Preface

Most of the contents of this thesis have been worked out during my employment at the Institute of Theoretical Computer Science, TU Braunschweig. I am grateful to everybody who contributed to make it such a pleasant and instructive time. The following acknowledgements address the many people who supported my work.

First of all, I thank my Ph.D. advisor Jirka Adámek for his guidance and enthusiasm as well as for sharing his great knowledge and experience. I thank Lutz Schröder who kindly assumed the function of the second examiner. I am grateful to Prof. Fekete for heading the Ph.D. committee.

I am very thankful to my colleague Stefan Milius for all collaboration, many discussions and hours in front of the whiteboard. I also thank all other institute members for always being open to scientific and administrative matters.

It was a pleasure to work together with Larry Moss on a paper. I am grateful to Thorsten Palm for collaboration on another paper. I enjoyed all the stays in Prague due to Jirka Velebils extraordinary skills in explaining category theory and his care for guests. Another thanks goes to Clemens Kupke for pointing Stefan, Larry and myself to the work of Silva and Rutten on infinite trees. I also thank all those who gave me valuable input and are not listed here.

This thesis would not have been possible without my family. This includes my parents who laid the foundation with their thoughtful and caring education. A special thanks is in order for the great support by my wife without which this thesis would not have been finished.

I am grateful for further support by Caren Buchmüller for carefully reading the thesis and improving my English; the remaining odd formulations are still mine. I thank my current employer, the Institute of Transportation Systems of the German Aerospace Center (DLR), for supporting my visit of the CSL conference in Bergen. And I enjoyed using Leslie Lamport's LaTeX, based on TeX by Donald Knuth, to write this thesis as well as other publications. For typesetting commutative diagrams, Kristoffer H. Rose's and

Ross Moore's xy-pic package was of great value, as well as the TikZ/PFG packages by Till Tantau for creating graphics.

Chapter 1

Introduction

The thesis at hand has one central conceptual theme: *recursive specification formats*. It has, however, two separate main parts in which two kinds of such formats are investigated. Part I is concerned with *compositional* formats; Part II focuses on *effectful* formats.

There also exists a recurrent technical theme in our thesis: *distributive laws*. More precisely, we use the concept of a distributive law between functors (possibly with some extra structure) from category theory, see [Bec69, TP97, LPW00]. According to the two main parts, we can distinguish two flavors of such distributive laws in the thesis. In Part I, algebraic operations are distributed over coalgebraic behavior whereas in Part II algebraic operations are distributed over effects.

This first chapter is intended as an easy access to the thesis. An introduction to recursion is given in Section 1.1. At the same time, the reader becomes familiar with basic vocabulary used throughout the thesis. In Section 1.2 an overview of the structure of this thesis and of the main results is given. Finally, in Section 1.3 important related work is presented.

The mathematical concepts utilized in the thesis stem from category theory (see e. g. [Mac98]) as well as from the theory of algebras and coalgebras for a functor (see e. g. [JR97, Rut00]). Their detailed introduction is postponed to Chapter 2.

1.1 Recursion

Recursion (from lat. recurrere = to run back, to return) is an important concept in mathematics and computer science. The great power of *recursive definitions* (or *recursive specifications*) rests on the ability to state that at some point a "similar situation as before" has occurred—instead of going on

with a long explicit definition. But it is not only the elegant, compact way of formulation that makes recursive definitions a successful tool: recursion is also a very natural and beautiful phenomenon. For example, the growth of shells can be described using recursion, and fractals arise from recursive definitions.

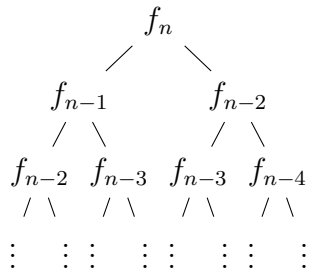
Among the most famous recursive definitions is the definition

$$f_0 = 0 \qquad f_1 = 1 \qquad f_n = f_{n-1} + f_{n-2} \quad (n \geq 2) \tag{1.1}$$

of the Fibonacci numbers: it states that the first two Fibonacci numbers are 0 and 1, and that otherwise the n-th Fibonacci number f_n is the sum of the previous two numbers. Using (1.1) repeatedly, we obtain the sequence $0, 1, 1, 2, 3, 5, 8, \ldots$. What makes the definition in (1.1) recursive is the right-hand equation: it is self-similar in the sense that a Fibonacci number is defined in terms of Fibonacci numbers.

Issues with Recursive Definitions

It is sometimes not easy to find a recursive formulation of a problem one would like to solve. Due to the compactness of such formulations little changes may cause a great difference in what is defined. Even if a recursive definition is found, it immediately gives rise to several questions: is it a real definition, i. e. does it define a unique object? How can this object be computed efficiently, and does the computation terminate? In our above example (1.1) of the Fibonacci numbers, the computation of a number f_n depends on the computation of f_{n-1} and f_{n-2} and so on as shown in the following tree:



We see that the computation terminates since in every recursion step one refers to numbers with strictly smaller indices, which finally leads back to the base cases f_0 and f_1. The efficiency of the naive approach to compute the number in every node of the tree can be improved by storing and reusing numbers that have been already computed; indeed, some numbers occur several times in the tree.

Whereas such computational issues need to be considered for the implementation of recursive definitions, we only included them here to "see how recursion works". In the thesis at hand we shall concentrate on the more fundamental question whether a recursive definition defines a unique object, or, as we shall say, whether it has a unique *solution*. Again we come back to our above example recursive definition of the Fibonacci numbers to see that it has a unique solution indeed: the sequence $0, 1, 1, 2, 3, 5, 8, \ldots$ is easily checked to satisfy the three defining equations in (1.1) and thus is a solution; the uniqueness follows from the fact that the first two numbers are given and all other numbers are computed step by step in a unique way from previous numbers in the sequence.

However, it is by no means obvious that a recursive definition has a unique solution. For example, consider the equation

$$x = 2/x \tag{1.2}$$

as a recursive definition. It has no solution in the set of natural numbers, and it has the two solutions $x = \sqrt{2}$ and $x = -\sqrt{2}$ in the set of real numbers. This shows us three important things we always must keep in mind when proving a recursive definition to have a unique solution: first, we need to fix a solution space together with an interpretation of the operation symbols occurring in the recursive definition (like / in (1.2)) on this solution space, i. e. we need an *algebra* in which the solution is taken. Second, we must show that there exists a solution. And third, uniqueness of this solution has to be proved.

Recursive Equations and Recursive Program Schemes

Classically, our solution space will be a set. We shall distinguish between recursive definitions defining elements (or *constants*) of this set and those defining functions (or *operations*) on it. We call the former recursive definitions *(systems of) recursive equations*, and the latter ones we call *recursive program schemes*. Our example (1.1) above defines a sequence and is—strictly speaking—neither a system of recursive equations nor a recursive program scheme. However, it is easy to regard this sequence as a stream which is an element from the set $\mathbb{R}^\omega$ or as a function on the set $\mathbb{N}$ of natural numbers. Thus we can illustrate both concepts by giving two further definitions of the Fibonacci numbers.

First, consider the following system of recursive equations:

$$\begin{aligned} x &= 0.y \\ y &= 1.(y + x) \end{aligned} \tag{1.3}$$

We solve it in the algebra carried by the set $\mathbb{R}^\omega$ of streams where $+$ is interpreted as the componentwise addition

$$[r_0, r_1, r_2, \dots] + [s_0, s_1, s_2, \dots] = [r_0 + s_0, r_1 + s_1, r_2 + s_2, \dots] \tag{1.4}$$

of streams $[r_0, r_1, r_2, \dots], [s_0, s_1, s_2, \dots] \in \mathbb{R}^\omega$ and $r.-$ as prefixing a stream $[r_0, r_1, r_2, \dots] \in \mathbb{R}^\omega$ with the real number r as in

$$r.[r_0, r_1, r_2, \dots] = [r, r_0, r_1, r_2, \dots]\,. \tag{1.5}$$

Repeated substitution according to the equations from (1.3) and application of addition and prefixing of streams gives the unique solutions $[0, 1, 1, 2, 3, 5, 8, \dots]$ for x and $[1, 1, 2, 3, 5, 8, \dots]$ for y where the former is the desired stream of Fibonacci numbers.

Second, consider the following recursive program scheme:

$$\mathsf{fib}(x) = \mathsf{threecases}(x, \mathsf{zero}, \mathsf{one}, \mathsf{fib}(x - \mathsf{one}) + \mathsf{fib}(x - \mathsf{two})) \tag{1.6}$$

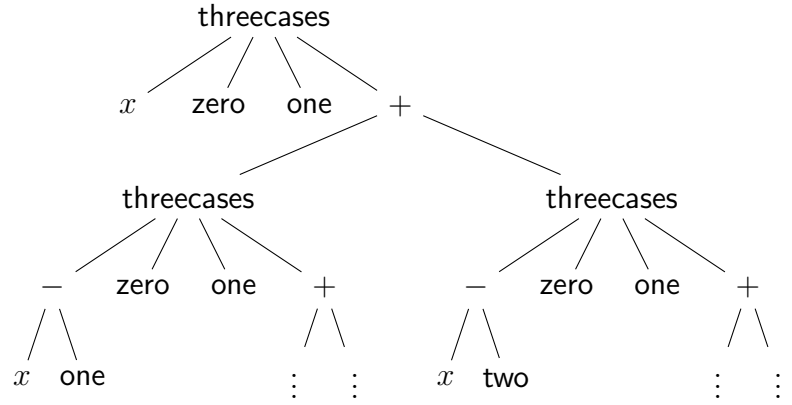
Here we choose the algebra given by the set $\mathbb{N}_\bot$ of natural numbers completed by an extra element $\bot$ and by the following interpretation of the occurring *given function symbols* (or *givens*, for short):

- zero, one and two are the constants 0, 1 and 2;
- $+$ is the usual addition with $n + \bot = \bot + n = \bot$;
- $-$ is the usual subtraction modified to $\bot$ for negative differences and with $n - \bot = \bot - n = \bot$; and
- $\mathsf{threecases}$ is the four-ary function

$$\mathsf{threecases}(n, m_0, m_1, m_{\text{else}}) = \begin{cases} \bot & n = \bot \\ m_0 & n = 0 \\ m_1 & n = 1 \\ m_{\text{else}} & \text{else}\,. \end{cases}$$

The unique solution of the unary *new function symbol* fib is the unary function $\mathsf{fib} : \mathbb{N}_\bot \to \mathbb{N}_\bot$ (which we again denote by fib) where $\mathsf{fib}(\bot) = \bot$ and $\mathsf{fib}(n)$ is given by the n-th Fibonacci number for every $n \in \mathbb{N}$.

Actually, the function $\mathsf{fib} : \mathbb{N}_\bot \to \mathbb{N}_\bot$ is the *interpreted solution* of the recursive program scheme (1.6) in $\mathbb{N}_\bot$. But we can also consider (1.6) without the interpretation in $\mathbb{N}_\bot$ as a purely syntactic construct. Then the *uninterpreted solution* of $\mathsf{fib}(x)$ is given by the infinite *tree* (or *term*)



which is obtained by unfolding the recursive definition (1.6).

So far, we assumed solution spaces to be sets, i. e. our algebras were carried by sets. In large parts of the thesis our category theoretic approach allows to generalize from the category of sets and functions to other categories where the solution space is an object of the category and the algebra is an algebra for a functor on that category. For simplicity, we also use the terms recursive definition (or recursive specification), (system of) recursive equations and recursive program scheme when working in different categories. Moreover, we also say—independent of the category—that systems of recursive equations define (or specify) constants and recursive program schemes define (or specify) operations (interpreted solution) and trees or terms (uninterpreted solution).

The following table summarizes the most important notions introduced in this section:

recursive definition/ recursive specification	solution
(system of) recursive equations	constants
recursive program scheme	operations (interpreted sol.)
	terms (uninterpreted sol.)

As we have seen above, solutions of recursive equations and interpreted solutions of recursive program schemes are always taken in an algebra.

In this thesis we provide different *formats* for recursive definitions and prove certain properties. Most often, we prove that recursive definitions of some format have a unique solution (possibly in a certain type of algebra). The term "format" refers to all kinds of restrictions and extensions one can impose on systems of recursive equations and recursive program schemes: for example only certain operations are allowed or are required in specific positions, or additional computational effects may be incorporated. It will

be best understood encountering concrete formats of recursive definitions in this thesis. Each format will be introduced with a concise definition of the format itself and the corresponding notion(s) of a solution.

1.2 Organization of this Thesis and Summary of Contributions

Organization

Our thesis has two main parts, both concerned with recursive specifications and both using techniques from category theory and coalgebra, in particular distributive laws and final coalgebras. Every part has its separate topic and is self-contained: Part I is concerned with compositionality of recursive specifications, and Part II deals with computational effects in recursive specifications. Both parts consist of an introduction to the topic and three chapters which provide the foundations for the part, formats of recursive equations and formats of recursive program schemes.

We describe the contents of Chapters 2–9 more detailed.

Chapter 2 sums up the basic facts and notation necessary to understand either of the two parts of this thesis.

Part I

Chapter 3. After recalling the concept of algebras for a pointed functor or monad and of coalgebras for a copointed functor in Section 3.1 we see how algebraic operations can be defined by distributive laws (Section 3.2) or by so called abstract GSOS rules (Section 3.3). Operations defined this way are then used in systems of recursive equations (Chapter 4) and in recursive program schemes (Chapter 5).

In **Chapter 4** we show in Section 4.1 that certain formats of recursive equations involving operations defined by abstract GSOS rules have unique solutions in final coalgebras. Moreover, those formats are compositional. We illustrate these results in Section 4.2 with the help of five concrete final coalgebras. Section 4.3 reviews three of these five examples asking what subsets of a final coalgebra can be defined using *finite* systems of recursive equations. Section 4.4 generalizes the results of Section 4.1 by using operations defined by distributive laws which are more general than abstract GSOS rules.

In **Chapter 5** we show in Section 5.1 that certain formats of recursive program schemes have unique interpreted solutions when interpreted in operations on a final coalgebra defined by abstract GSOS rules. Moreover, those formats are compositional. We illustrate these result in Section 5.2 on the five final coalgebras of Section 4.2.

Part II

Chapter 6. In Section 6.1 five monads are described which correspond to computational effects investigated throughout Part II. Section 6.2 provides canonical distributive laws of a substantial number of functors over each of these monads. Those distributive laws serve as "policies for effect-handling".

Chapter 7. Here we define in Section 7.1 modifications of completely iterative algebras (CIAs) of Milius [Mil05] for our setting with computational effects: Kleisli-CIAs and λ-CIAs, where λ is a given distributive law. For the five effects and their canonical distributive laws introduced in Chapter 6 we investigate the free algebras of that type in Section 7.2 and characterize them in Section 7.3.

In **Chapter 8** we are concerned with uninterpreted solutions of recursive program schemes with computational effects (using again the five effects from Chapter 6). To this end, we first need to extend the canonical distributive laws from Section 6.2 in Section 8.1. Section 8.2 is an intermediate step to the main result of Section 8.3 where certain structures are proved to be (weakly) final coalgebras. The latter is a key result in proving uninterpreted recursive program schemes to have unique (canonical) solutions which is done in Section 8.5 with another intermediate step in Section 8.4. Chapter 8 concludes Part II.

Finally **Chapter 9** summarizes our results and lists future work.

Contributions

The contributions of this thesis (including the results from our papers underlying the thesis, see Section 1.3) can be found in Chapters 4–8. We consider the following as our main achievements:

Theorems 4.7 and 4.8 were first published in our conference paper [MMS10] in a slightly different formulation and in its journal version [MMS13] in the present formulation. They state that initial CIAs can be extended by operations obtained from abstract GSOS rules in two ways to obtain new CIAs. Since a general compositionality principle [AMV06a] holds for CIAs, we conclude that our two new formats of recursive equations associated to the new CIAs, which use the additional operations, are compositional. Theorems 5.1, 5.16 and 5.27 originate from [MMS10, MMS13] as well and state that the unique interpreted solutions of three different formats of recursive program schemes (RPS's) in a CIA extend the latter to give a new CIA. It follows that the additional operations from this new CIA can be used in further RPS's of the three formats, which still have unique interpreted solutions. As was explained in loc. cit., our results imply and extend known results e. g. for Milner's calculus of communicating systems (CCS) [Mil89],

Rutten's stream calculus [Rut05a] and formal languages theory.

Theorems 7.47, 7.50 and 7.53 were first published in our paper [MPS09]. They state that if λ is a canonical distributive law (see the description of Chapter 6 above) which handles partial, nondeterministic or composite computations, Kleisli-CIAs and λ-CIAs (see the description of Chapter 7 above) coincide. Moreover, in each of the three cases of canonical distributive laws they provide a further characterization of them. Still in the context of underlying canonical distributive laws, Theorems 8.50 (a similar result was conjectured in our short contribution [Sch10b]), 8.60 (first proved in [Sch11]) and 8.63 (new in this thesis) deal with uninterpreted solutions of recursive program schemes with effects. For guarded partial, nondeterministic, probabilistic and composite RPS's the uniqueness of solutions is shown. For guarded nonempty nondeterministic RPS's, solutions are no longer unique in general, but it is shown that greatest solutions can be obtained in a canonical way.

In Chapters 2 and 3 only known material is recollected; Chapters 4–8 contain new results, all the results from the literature can be recognized from the references given in parentheses directly after the name of a theorem, proposition etc.

1.3 Related Work and the Author's Publications

In this section, we list the literature most influential for this thesis. We also mention previous publications of parts of the thesis. Notice that at the end of each chapter there is a separate related work section where more specific literature is discussed.

An important foundation for our work is given by the category theoretic notions of recursive equations and recursive program schemes and the connected results, cf. Section 2.3. Regarding recursive equations, our thesis continues the line of research whose cornerstones are Milius' paper [Mil05] and the papers [AMV06a, AMV06b] by Adámek, Milius and Velebil. For recursive program schemes, our work is influenced by the paper [GLM03] by Ghani, Lüth and De Marchi and especially by Milius' and Moss' paper [MM06]. The category theoretic notions of recursive equations and recursive program schemes were in turn inspired by corresponding classical notions—more details can be found in the papers cited in this paragraph.

Another important notion used throughout the thesis is distributive laws. This goes back to Beck [Bec69] who investigated distributive laws

of monads. The variants of distributive laws and the connected results that Part I of our thesis builds upon (cf. Section 3.2) can be found in Bartels' work [Bar03, Bar04]. The special case of abstract GSOS rules (cf. Section 3.3) was already treated by Turi and Plotkin [TP97] and in subsequent work by Lenisa, Power and Watanabe [LPW00, LPW04]. The canonical distributive laws in Part II (cf. Sections 6.2 and 8.1) continue the work of Hasuo, Jacobs and Sokolova [HJS07].

Parts of our thesis have been already published: the results of Part I in Sections 4.1, 4.4, and 5.1 stem from the joint work with Milius and Moss [MMS13] (the journal version of the conference paper [MMS10]). An exception are Remarks 4.15 and 4.70, Lemmata 5.13 and 5.14 and Proposition 5.30. Whereas the five example domains from Sections 4.2 and 5.2 are already in the papers, several of the concrete examples are not. Section 4.3 has not been previously published except in a remark summarizing some of the results for formal languages from Section 4.3.3.

In Part II, parts of Chapter 6 (in particular Theorem 6.19) and the whole Chapter 7 (except for the parts of Section 7.2 concerning the nonempty powerset and identity monads) were published without some of the proofs in the conference paper [MPS09] coauthored by Milius, Palm and ourselves. All results of Chapters 7 and 8 related to the nonempty powerset monad except Remark 8.38 and Examples 8.39 and 8.58 are from our conference paper [Sch11] where nearly all proofs were omitted. We also mention the short abstract [Sch10b] as a precursor to that paper. The results of Chapter 8 on RPS's with effects corresponding to monads other than the nonempty powerset monad have not yet been published; the same is true for Remark 8.38 and Examples 8.39 and 8.58.

Chapter 2

Preliminaries

In this chapter we introduce several mathematical concepts needed in both of the main parts of this thesis. Whenever we need a concept only in one of the two parts, we shall introduce it there.

We assume that the reader of this thesis is familiar with some basic concepts of category theory: we use the notions of category, functor and natural transformation without further explanation, as well as limit (in particular product and pullback) and colimit (in particular coproduct). We shall also widely make use of commutative diagrams. For an introduction to category theory, the reader is referred to the literature, see Section 2.4 below.

Further concepts from category theory needed in this thesis are explained in Section 2.1 of this chapter; among those is the concept of a distributive law which is present throughout the whole thesis, and further concepts as polynomial set functors, (co-)pointed functors and monads. Section 2.2 introduces algebras and coalgebras for a functor. The probably less well-known notions of completely iterative algebras, complete Elgot algebras, completely iterative monads and of (category theoretical) recursive program schemes are presented in Section 2.3. These are of great importance for us since we will encounter several variations throughout this work.

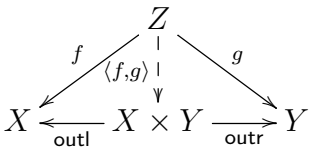
For most readers with some background in category theory and coalgebra, Section 2.3 should be a good starting point; in this case Sections 2.1 and 2.2 may serve to look up notation when necessary.

As a first notational convention please notice that throughout this thesis $\mathbb{N}$ denotes the natural numbers including 0.

2.1 Notions from Category Theory

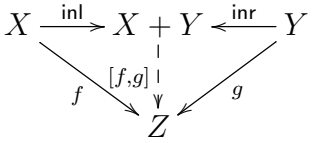
Notation for Morphisms and Natural Transformations

For binary products $X \times Y$, the *left and right projections* are denoted by $\mathsf{outl} : X \times Y \to X$ and $\mathsf{outr} : X \times Y \to Y$. The *pairing* $\langle f, g \rangle : Z \to X \times Y$ of two morphisms $f : Z \to X$ and $g : Z \to Y$ is the unique morphism making the diagram



commute.

For binary coproducts $X + Y$, the *left and right injections* are denoted by $\mathsf{inl} : X \to X + Y$ and $\mathsf{inr} : Y \to X + Y$. The *copairing* $[f, g] : X + Y \to Z$ of two morphisms $f : X \to Z$ and $g : Y \to Z$ is the unique morphism making the diagram



commute.

Given a functor G, we denote by can the canonical morphism $[G\mathsf{inl}, G\mathsf{inr}] : GX + GY \to G(X + Y)$.

We also use the notations inl, inr and can for natural transformations $\mathsf{inl} : H \to H + H'$, $\mathsf{inr} : H' \to H + H'$ and $\mathsf{can} : GH + GH' \to G(H + H')$ for functors H, H' and G with components given by the respective morphisms inl, inr and can. We denote *parallel composition* of natural transformations $\alpha : G \to G'$ and $\beta : H \to H'$ by $\alpha * \beta : GH \to G'H'$.

Classification of Functors

On some occasions we use the terms *finitary functor* and *accessible functor*. For a general formal definition we refer the reader to the book [AR94]; here we only give a characterization of these terms for endofunctors on the category Set of sets and functions (or *set functors*, for short), see Definition 2.5 below.

In most examples of Part I of this thesis and throughout Part II we shall work with polynomial endofunctors on Set which we define next.

Definition 2.1. A *signature* Σ is a set of operation symbols where each operation symbol $\sigma \in \Sigma$ has an associated cardinal n_σ, its *arity*. Σ is called

k-ary for some infinite cardinal k if $n_\sigma < k$ for all $\sigma \in \Sigma$. In particular, an ω-ary signature is also called *finitary*.

Definition 2.2. A *polynomial set functor* for some signature Σ is a functor of the form $\coprod_{\sigma\in\Sigma} \mathrm{Id}^{n_\sigma}$.

Notation 2.3. Given a signature Σ, we let $\Sigma_n = \{\sigma \in \Sigma \mid n_\sigma = n\}$. Observe that a polynomial functor for a k-ary signature Σ equivalently is a functor of the form $\coprod_{n<k} \Sigma_n \times \mathrm{Id}^n$. Occasionally we write H_Σ for a polynomial functor for the signature Σ. Elements from $H_\Sigma X$ are denoted by $\sigma(x_1, \ldots, x_{n_\sigma})$ where $x_1, \ldots, x_{n_\sigma} \in X$.

In conjunction with signatures and polynomial functors we need the following

Definition 2.4. Given a signature Σ, a Σ-*tree* is a rooted, ordered tree with node labels in Σ where every node labeled by $\sigma \in \Sigma$ has precisely n_σ children. If additionally a variable set X is given, a Σ-*tree on* X is a $(\Sigma+X)$-tree where to each variable from X the arity 0 is assigned, i. e. leaves may be labeled not only by constants from Σ but also by variables from X.

Definition 2.5. Given a regular cardinal k, a set functor H is *k-accessible* if it is a quotient of some polynomial functor H_Σ for a k-ary signature Σ (see [AP04] for further equivalent conditions). H is *accessible* if it is k-accessible for some regular cardinal k; and H is *finitary* if it is ω-accessible.

Thus for set functors we have the following inclusions

$$\begin{array}{ccc} \text{finitary} & \subset & \text{accessible} \\ \cup & & \cup \\ \begin{array}{c}\text{finitary}\\ \text{polynomial}\end{array} & \subset & \text{polynomial} \end{array}$$

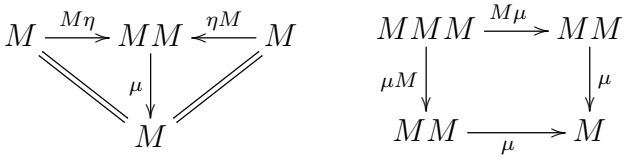
where the finitary polynomial functors are precisely the polynomial functors for a finitary signature.

(Co-)Pointed Functors and Monads

Throughout the thesis we shall add functors as superscripts of natural transformations whenever we need to distinguish natural transformations of similar function that are otherwise denoted by the same letter. We shall omit such superscripts whenever confusion is unlikely.

Definition 2.6. A *pointed functor* (M, η) is an endofunctor M together with a natural transformation $\eta : \mathrm{Id} \to M$, the *unit*.

Definitions 2.7. 1. A *monad* (M, η, μ) is an endofunctor M together with a natural transformation $\eta : \mathrm{Id} \to M$, the *unit*, and a natural transformation $\mu : MM \to M$, the *multiplication*, such that the following diagrams commute:



The two axioms on the left are called the *unit laws*, and the axiom on the right is called the *multiplication law* of the monad.

2. A *monad morphism* between monads (M, η^M, μ^M) and (N, η^N, μ^N) is a natural transformation $\theta : M \to N$ such that $\theta \cdot \eta^M = \eta^N$ and $\theta \cdot \mu^M = \mu^N \cdot N\theta \cdot \theta M$.

Definition 2.8. A *free monad* on an endofunctor H is a monad (F^H, η^H, μ^H) together with a natural transformation $\kappa : H \to F^H$, called *universal*, such that for every monad (M, η^M, μ^M) together with a natural transformation $\alpha : H \to M$ there is a unique monad morphism $\alpha^\# : F^H \to M$ with $\alpha^\# \cdot \kappa = \alpha$. We say that $\alpha^\#$ *extends* α.

Example 2.9. For free monads on polynomial set functors H_Σ, $F^{H_\Sigma}X$ can be viewed as the set of all Σ-terms or Σ-trees (of finite depth) over variables from X. $\eta_X^{H_\Sigma}$ makes variables from X into terms or trees of depth 0, $\mu_X^{H_\Sigma}$ performs "term flattening" or "tree flattening" and κ_X is the injection of "flat" terms or trees of depth at most 1 into the set of all terms.

Definition 2.10. A *copointed functor* (D, ε) is an endofunctor D together with a natural transformation $\varepsilon : D \to \mathrm{Id}$, the *counit*.

Definition 2.11. A *cofree copointed functor* on an endofunctor H is a copointed functor (Q^H, ε^H) together with a natural transformation $\iota : Q^H \to H$, such that for every copointed functor (D, ε^D) together with a natural transformation $\alpha : D \to H$ there is a unique natural transformation $\alpha^\# : D \to Q^H$ with $\varepsilon^H \cdot \alpha^\# = \varepsilon^D$ and $\iota \cdot \alpha^\# = \alpha$.

Example 2.12. If we work in a category with binary products, the cofree copointed functor on a functor H simply is $(H \times \mathrm{Id}, \mathsf{outr})$ together with $\mathsf{outl} : H \times \mathrm{Id} \to H$.

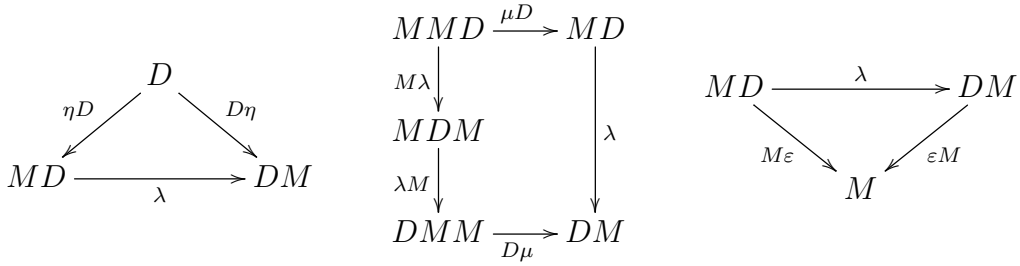
Distributive Laws

Distributive laws go back to Beck [Bec69] who investigated distributive laws of monads (see Definition 2.15 below). Since then many variants of distributive laws have been investigated of which the following is the most basic one.

Definition 2.13. A *distributive law of endofunctors* G and G' is a natural transformation $\lambda : GG' \to G'G$. We say that G *distributes over* G'.

Variants of distributive laws arise if the functors G and G' carry the additional structure of (co-)pointed functors or monads, for example. For each natural transformation which is part of such structures, the distributive law λ is then required to satisfy an obvious law expressing compatibility of the natural transformation and λ. Below we introduce some variants of distributive laws that are of interest in this thesis.

Definition 2.14. A *distributive law of a monad* (M, η, μ) *over a copointed functor* (D, ε) is a natural transformation $\lambda : MD \to DM$ such that the following diagrams commute:



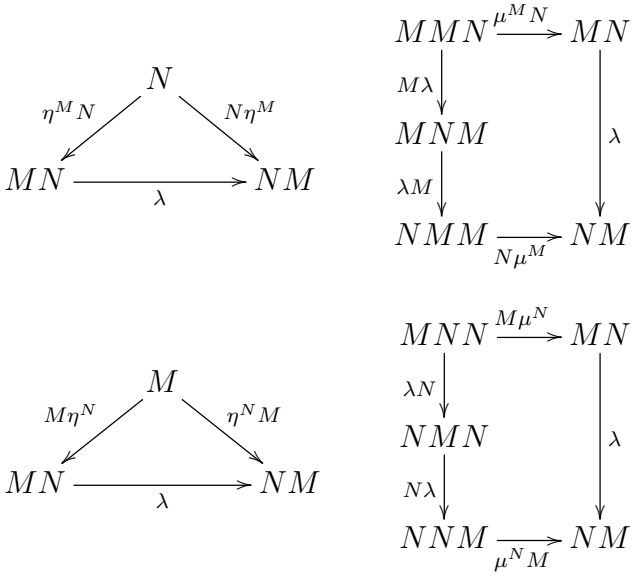
These are called the *unit law*, the *multiplication law* and the *counit law* for λ, respectively.

Leaving out the respective natural transformations and laws, one obtains the notions of

- a *distributive law of a monad over a functor*,
- a *distributive law of a pointed functor over a functor*, and
- a *distributive law of a pointed functor over a copointed functor*.

The notion of a distributive law of a functor over a copointed functor can also be obtained but is not needed in our work.

Definition 2.15 ([Bec69])**.** A *distributive law of monads* (M, η^M, μ^M) and (N, η^N, μ^N) is a natural transformation $\lambda : MN \to NM$ such that the following diagrams commute:



The triangles are called the *unit laws* for λ, and the other two diagrams are called the *multiplication laws* for λ.

Viewing M as a mere functor and leaving out the laws given by the two upper diagrams, we obtain the notion of a *distributive law of a functor over a monad*.

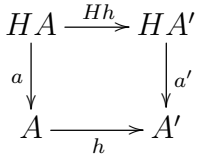
2.2 Algebras and Coalgebras

Algebras

Definitions 2.16.

1. An *algebra for an endofunctor* H, or H*-algebra* for short, is an object A together with a morphism $a : HA \to A$. When convenient, we may denote this by (A, a) or just mention $a : HA \to A$ since the object A is already given as part of the morphism.

2. A *homomorphism* between H-algebras (A, a) and (A', a') is a morphism $h : A \to A'$ such that the following diagram commutes:



All H-algebras together with the H-algebra homomorphisms between them form a category H-Alg.

Example 2.17. Classical Σ-algebras are special instances of the above definition where $H = H_\Sigma$ is a polynomial set functor for a finitary signature. In fact, consider for example an algebra with a constant $a_0 \in A$ and a binary operation $\sigma^A : A \times A \to A$. It is an algebra for the functor $H_\Sigma X = \{*\} + X \times X$ by considering the constant as a map $con : \{*\} \to A$ with $* \mapsto a_0$ and taking the copairing $[con, \sigma^A] : \{*\} + A \times A \to A$.

Homomorphisms between H_Σ-algebras are the classical Σ-algebra homomorphisms, i. e. structure preserving maps.

Definition 2.18. An *initial H-algebra* is an H-algebra (I, i) such that for every H-algebra (A, a) there is a unique homomorphism between (I, i) and (A, a).

Definition 2.19. A *free H-algebra* on an object X is an H-algebra (FX, ϕ_X) together with a morphism $\eta_X : X \to FX$, called *universal*, such that for every H-algebra (A, a) together with a morphism $f : X \to A$ there is a unique homomorphism $f^\# : FX \to A$ with $f^\# \cdot \eta_X = f$. We say that $f^\#$ *extends* f.

Remarks 2.20.

1. By Lambek's lemma [Lam68], $i : HI \to I$ is an isomorphism for every initial H-algebra (I, i).

2. If H is a functor on a category with coproducts, the following are equivalent:

   (a) (FX, ϕ_X) is the free H-algebra on X with universal morphism η_X.

   (b) $(FX, [\phi_X, \eta_X])$ is the initial $(H(-) + X)$-algebra.

   From item (1) we conclude that $[\phi_X, \eta_X]$ is an isomorphism.

3. Initial and free algebras need not exist, but if they do, they are unique up to isomorphism. For this reason we often write "the initial/free algebra".

Example 2.21. The free algebra $\phi_X : H_\Sigma FX \to FX$ on X for a polynomial set functor H_Σ with universal morphism $\eta_X : X \to FX$ is given by the set FX of all finite Σ-trees on X, "tree tupling"

$$\phi_X(\sigma(t_1, \ldots, t_n)) = \begin{array}{c} \sigma \\ /\,|\,\backslash \\ t_1 \cdots t_n \end{array}$$

for $\sigma \in \Sigma_n$ and $t_1, \ldots, t_n \in FX$, and the singleton tree $\eta_X(x)$ labeled by x for every $x \in X$.

The following theorem connects free algebras and free monads and justifies our notation of free algebras which look like components of a natural transformation:

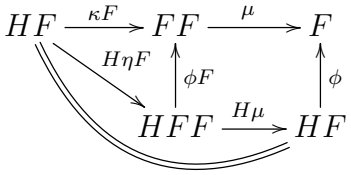
Theorem 2.22 ([Bar70], see also [Kel80])**.** *If for every object X the free H-algebra (FX, ϕ_X) on X with universal morphism η_X exists, the free monad (F, η, μ) on H with universal natural transformation κ is given as follows:*

- *the functor F is defined on objects X by FX and on morphisms $f : X \to Y$ by the unique homomorphism between (FX, ϕ_X) and (FY, ϕ_Y) extending $\eta_Y \cdot f$;*
- *the components of η are given by η_X for every X;*
- *the components of μ are the unique homomorphisms between (FFX, ϕ_{FX}) and (FX, ϕ_X) extending id_{FX}; and*
- *the components of κ are given by $\phi_X \cdot H\eta_X$ for every X.*

Remark 2.23. From the proof of Theorem 2.22 we have that the ϕ_X form a natural transformation ϕ. From the definition of μ we obtain $\mu_X \cdot \phi_{FX} = \phi_X \cdot H\mu_X$ for every object X; together this means $\mu \cdot \phi F = \phi \cdot H\mu$.

Lemma 2.24. *It holds $\phi = \mu \cdot \kappa F$.*

Proof. Consider the following diagram:

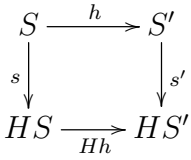


The triangles are the definition of κ and one of the monad unit laws, and for the right-hand square see Remark 2.23. Thus the desired outer square commutes. □

Coalgebras

Definitions 2.25. 1. A *coalgebra for an endofunctor H*, or *H-coalgebra* for short, is an object S together with a morphism $s : S \to HS$. When convenient, we may denote this by (S, s) or just mention $s : S \to HS$ since the object S is already given as part of the morphism.

2. A *homomorphism* between H-coalgebras (S, s) and (S', s') is a morphism $h : S \to S'$ such that the following diagram commutes:



All H-coalgebras together with the H-coalgebra homomorphisms between them form a category H-Coalg.

Example 2.26. Coalgebras (S, s) for set functors can be viewed as state-based systems: S is the set of states, and s gives the structure of the system (one-step transitions between states). The functor determines the system type. For example, coalgebras for the functor $HX = X^A \times 2$ are precisely the deterministic automata with input alphabet A (without an initial state); and coalgebras for $HX = \mathcal{P}(A \times X)$ are precisely the (possibly infinitely branching) labeled transition systems with labels from A.

Homomorphisms between H-coalgebras preserve the transition structure.

Definition 2.27. A *final H-coalgebra* is an H-coalgebra (C, c) such that for every H-coalgebra (S, s) there is a unique homomorphism between (S, s) and (C, c).

Remarks 2.28.

1. By the dual of Lambek's lemma [Lam68], $c : C \to HC$ is an isomorphism for every final H-coalgebra (C, c).

2. Final coalgebras need not exist, but if they do, they are unique up to isomorphism. For this reason we often write "the final coalgebra".

Example 2.29. The final coalgebra $c : TX \to H_\Sigma TX + X$ for $H_\Sigma(-) + X$, where H_Σ is a polynomial set functor, is given by the set TX of all finite and infinite Σ-trees on X and the inverse of "tree tupling" (cf. Example 2.21): c maps a singleton tree labeled by x to x, and a tree with root label $\sigma \in \Sigma_n$ and child trees $t_1, \ldots, t_n \in TX$ to $\sigma(t_1, \ldots, t_n)$.

2.3 Category Theoretical Recursive Specifications

Completely Iterative Algebras

Definition 2.30 ([Mil05])**.** Let H be an endofunctor on a category with coproducts. A *flat equation morphism* (with parameters in A) is a morphism

$e : X \to HX + A$. A *solution* of e in an H-algebra (A, a) is a morphism $e^\dagger : X \to A$ such that the diagram

$$\begin{array}{ccc} X & \xrightarrow{e^\dagger} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [a,\mathrm{id}_A]} \\ HX + A & \xrightarrow[He^\dagger + \mathrm{id}_A]{} & HA + A \end{array} \tag{2.1}$$

commutes. A *completely iterative algebra for* H (or H*-CIA*, for short) is an H-algebra (A, a) in which every flat equation morphism with parameters in A has a unique solution.

All H-CIAs together with the H-algebra homomorphisms between them form a category H-CIA.

For one of the following Examples 2.32 of CIAs (and also in Chapter 7) we shall need the following definition:

Definition 2.31. A (strict partial) order $<$ on a set X is called *well-founded* if there is no infinite descending chain $x_0 > x_1 > x_2 > \dots$ where $x_i \in X$ for all $i \in \mathbb{N}$. Similarly, a directed graph is called well-founded if it has no infinite path (the connection of the two concepts being that the ordering decreases in the direction of the edges).

Examples 2.32 (cf. [Mil05])**.**

1. Let $c : TX \to HTX + X$ denote a final coalgebra for the functor $H(-)+X$. The morphism c is an isomorphism by Remark 2.28(1), and so its inverse yields an H-algebra $\tau_X = c^{-1} \cdot \mathsf{inl} : HTX \to TX$ and a morphism $\eta_X = c^{-1} \cdot \mathsf{inr} : X \to TX$. Then (TX, τ_X) is a free CIA on X with universal morphism η_X, see Corollary 2.35 below.

2. Let H_Σ be a polynomial set functor. Recall the final coalgebra $c : TX \to H_\Sigma TX + X$ for $H_\Sigma(-) + X$ from Example 2.29. Together with item (1) we conclude that the free CIA $\tau_X : H_\Sigma TX \to TX$ for H_Σ on X with universal morphism $\eta_X : X \to TX$ is carried by the set TX of finite and infinite Σ-trees on X, that τ_X performs tree tupling and η_X considers elements $x \in X$ as singleton trees labeled by x.

3. The algebra of addition on $\{1, 2, 3, \dots\} \cup \{\infty\}$ is a CIA for the set functor $HX = X \times X$.

4. Unary algebras of Set are algebras for the set functor $H = \mathrm{Id}$. A unary algebra (A, a) is a CIA iff $a : A \to A$ has a fixed point $a_0 \in A$ and there is no infinite sequence $a_1, a_2, a_3, \dots$ with $a_i = a(a_{i+1})$, $i = 1, 2, 3, \dots$,

except for the one all of whose members are a_0. The second part of this condition can be put more vividly as follows: the graph with node set $A \setminus \{a_0\}$ and with an edge from $a(b) \neq a_0$ to b for all $b \in A$ is well-founded. Since any well-founded graph induces a well-founded (strict) order on its node set, we have yet another formulation: there is a well-founded order on $A \setminus \{a_0\}$ for which $a : A \to A$ is strictly increasing in the sense that $a(b) \neq a_0$ implies $b < a(b)$ for all $b \in A$.

5. Classical algebras are seldom CIAs. For example, a group or a semilattice is a CIA (for $HX = X \times X$) iff they contain one element only (consider the unique solution of $x = x \cdot 1$ or $x = x \vee x$, respectively).

For further examples we refer the reader to [Mil05].

Theorem 2.33 ([Mil05])**.** *The following are equivalent:*

1. (C, c) *is the final* H*-coalgebra.*
2. (C, c^{-1}) *is the initial* H*-CIA.*

Remarks 2.34. 1. The inverse of the final coalgebra morphism $c : C \to HC$ exists since by Remark 2.28(1) c is an isomorphism.

2. Similar to Remark 2.20(2), the following are equivalent (see [Mil05], Theorem 2.10):
   (a) (TX, τ_X) is the free H-CIA on X with universal morphism η_X.
   (b) $(TX, [\tau_X, \eta_X])$ is the initial $(H(-) + X)$-CIA.

From Theorem 2.33 and Remark 2.34(2) we obtain

Corollary 2.35 ([Mil05])**.** *The following are equivalent:*

1. $(TX, [\tau_X, \eta_X]^{-1})$ *is the final* $(H(-) + X)$*-coalgebra.*
2. (TX, τ_X) *is the free* H*-CIA on* X *with universal morphism* η_X*.*

Complete Elgot Algebras

Notation 2.36. Let $e : X \to HX + Y$ and $f : Y \to HY + A$ be flat equation morphisms and let $h : Y \to Z$ be any morphism. We denote by $h \bullet e$ and $f \blacksquare e$ the flat equation morphisms

$$h \bullet e \equiv (X \xrightarrow{e} HX + Y \xrightarrow{HX+h} HX + Z)$$

(which is e with parameters renamed by h) and

$$f \blacksquare e \equiv (X + Y \xrightarrow{[e,\mathsf{inr}]} HX + Y \xrightarrow{HX+f} HX + HY + A \xrightarrow{\mathsf{can}+A} H(X + Y) + A)$$

(which is the composite of e and f).

Definition 2.37. Let $a : HA \to A$ be an H-algebra. A function $(-)^\dagger$ assigning to each flat equation morphism $e : X \to HX + A$ with parameters in A a solution $e^\dagger : X \to A$ in (A, a) is called

- *functorial* if for every homomorphism $h : X \to Y$ between flat equation morphisms $e : X \to HX + A$ and $f : Y \to HY + A$ (i. e. $(Hh + A) \cdot e = f \cdot h$) we have $e^\dagger = f^\dagger \cdot h$;
- *compositional* if for any flat equation morphisms $e : X \to HX + Y$ and $f : Y \to HY + A$ we have $(f \blacksquare e)^\dagger \cdot \mathsf{inl} = (f^\dagger \bullet e)^\dagger$.

Definitions 2.38 ([AMV06a])**.** Let H be an endofunctor on a category $\mathcal{C}$ with coproducts.

1. A *complete Elgot algebra for H* (or *H-CEA*, for short) is an H-algebra $a : HA \to A$ together with a function $(-)^\dagger$ assigning to each flat equation morphism $e : X \to HX + A$ with parameters in A a solution $e^\dagger : X \to A$ in (A, a) such that $(-)^\dagger$ is functorial and compositional.
2. A morphism $h : A \to B$ in $\mathcal{C}$ between H-CEAs $(A, a, (-)^\dagger)$ and $(B, b, (-)^\ddagger)$ is called *solution preserving* if for all flat equation morphisms $e : X \to HX + A$ the equation $h \cdot e^\dagger = (h \bullet e)^\ddagger$ holds.

All H-CEAs together with the solution preserving morphisms between them form a category H-CEA. Notice that solution preserving morphisms automatically are algebra homomorphisms as proved in [AMV06a], Lemma 4.3.

Examples 2.39. 1. Every H-CIA is an H-CEA. In fact, it follows from [Mil05, AMV06a] that H-CIA is a full subcategory of H-CEA.

2. Continuous algebras on CPOs are CEAs, see Example 3.4 in [AMV06a], but they are no CIAs in general.

For further examples of CEAs, we refer the reader to [AMV06a].

Theorem 2.40 ([AMV06a], Theorem 5.4)**.** *The following are equivalent:*

1. *$\tau_X : HT^H X \to T^H X$ is the free H-CEA on X with universal arrow $\eta_X : X \to T^H X$;*
2. *$[\tau_X, \eta_X]^{-1} : T^H X \to HT^H X + X$ is the final $(H(-) + X)$-coalgebra.*

Together with Corollary 2.35 we conclude that the free H-CEAs are precisely the free H-CIAs.

Completely Iterative Monads

Definitions 2.41. 1. Given a monad (M, η, μ), a *(right) M-module* (G, ν) is an endofunctor G together with a natural transformation $\nu : GM \to G$ such that the following diagrams commute:

$$\begin{array}{ccc} G & \xrightarrow{G\eta} & GM \\ & \searrow^{=} & \downarrow{\scriptstyle \nu} \\ & & G \end{array} \qquad \begin{array}{ccc} GMM & \xrightarrow{G\mu} & GM \\ {\scriptstyle \nu M}\downarrow & & \downarrow{\scriptstyle \nu} \\ GM & \xrightarrow[\nu]{} & G \end{array}$$

2. A *module homomorphism* between M-modules (G, ν^G) and $(G', \nu^{G'})$ is a natural transformation $\vartheta : G \to G'$ such that the following diagram commutes:

$$\begin{array}{ccc} GM & \xrightarrow{\vartheta M} & G'M \\ {\scriptstyle \nu^G}\downarrow & & \downarrow{\scriptstyle \nu^{G'}} \\ G & \xrightarrow[\vartheta]{} & G' \end{array}$$

Example 2.42. For every functor G and monad (M, η, μ), the pair $(GM, G\mu)$ is an M-module. Indeed, this follows from the monad laws of M. In particular, $G = \mathrm{Id}$ yields that (M, μ) is an M-module for every monad (M, η, μ).

Definitions 2.43. 1. An *idealized monad* $(M, \eta, \mu, \bar{M}, \bar{\mu}, \vartheta)$ is a monad (M, η, μ) together with an M-module $(\bar{M}, \bar{\mu})$ and a module homomorphism $\vartheta : \bar{M} \to M$ between $(\bar{M}, \bar{\mu})$ and (M, μ).

2. An *ideal natural transformation* is a natural transformation $\alpha : G \to M$ into an idealized monad $(M, \eta, \mu, \bar{M}, \bar{\mu}, \vartheta)$ that factors as follows:

$$\alpha \equiv (G \xrightarrow{\bar{\alpha}} \bar{M} \xrightarrow{\vartheta} M)\,.$$

3. An *idealized monad morphism* $(\theta, \bar{\theta})$ between idealized monads $(M, \eta^M, \mu^M, \bar{M}, \bar{\mu}^M, \vartheta^M)$ and $(N, \eta^N, \mu^N, \bar{N}, \bar{\mu}^N, \vartheta^N)$ is a monad morphism $\theta : M \to N$ together with a natural transformation $\bar{\theta} : \bar{M} \to \bar{N}$ such that the following diagrams commute:

$$\begin{array}{ccc} \bar{M}M & \xrightarrow{\bar{\theta} * \theta} & \bar{N}N \\ {\scriptstyle \bar{\mu}^M}\downarrow & & \downarrow{\scriptstyle \bar{\mu}^N} \\ \bar{M} & \xrightarrow[\bar{\theta}]{} & \bar{N} \end{array} \qquad \begin{array}{ccc} \bar{M} & \xrightarrow{\bar{\theta}} & \bar{N} \\ {\scriptstyle \vartheta^M}\downarrow & & \downarrow{\scriptstyle \vartheta^N} \\ M & \xrightarrow[\theta]{} & N \end{array}$$

Example 2.44. Every monad (M, η, μ) can be canonically completed to an idealized monad $(M, \eta, \mu, M, \mu, \mathrm{id})$. In general, there are other ways to complete M to an idealized monad, see e. g. Theorem 2.47 below.

Definition 2.45 ([Mil05])**.** Let $(M, \eta, \mu, \bar{M}, \bar{\mu}, \vartheta)$ be an idealized monad. An *equation morphism* (with parameters in Y) is a morphism $e : X \to M(X+Y)$. It is called *guarded* if it factors as

$$e \equiv (X \xrightarrow{e'} \bar{M}(X+Y) + Y \xrightarrow{[\vartheta_{X+Y}, \eta_{X+Y} \cdot \mathsf{inr}]} M(X+Y))$$

for some morphism $e' : X \to \bar{M}(X+Y)+Y$. A *solution* of e with parameters in Y is a morphism $e^\dagger : X \to MY$ such that the diagram

$$\begin{array}{ccc} X & \xrightarrow{e^\dagger} & MY \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle \mu_Y} \\ M(X+Y) & \xrightarrow[M[e^\dagger, \eta_Y]]{} & MMY \end{array}$$

commutes. An idealized monad is called *completely iterative* (or *CIM*, for short) if every guarded equation morphism has a unique solution.

Definition 2.46. A *free CIM* on an endofunctor H is a CIM $(T^H, \eta^H, \mu^H, \bar{T}^H, \bar{\mu}^H, \vartheta^H)$ together with an ideal natural transformation $\kappa : H \to T^H$, called *universal*, such that for every CIM $(M, \eta^M, \mu^M, \bar{M}, \bar{\mu}^M, \vartheta^M)$ and every ideal natural transformation $\alpha : H \to M$ there is a unique idealized monad morphism $(\alpha^\# : T^H \to M, \bar{\alpha}^\# : \bar{T}^H \to \bar{M})$ such that $\bar{\alpha}^\# \cdot \bar{\kappa} = \bar{\alpha}$ (and thus $\alpha^\# \cdot \kappa = \alpha$). We say that $\alpha^\#$ and $\bar{\alpha}^\#$ *extend* α and $\bar{\alpha}$, respectively.

Theorem 2.47 ([Mil05])**.** *If for every object X the free H-CIA (TX, τ_X) on X with universal arrow η_X exists, the free CIM $(T, \eta, \mu, \bar{T}, \bar{\mu}, \vartheta)$ on H with universal natural transformation κ is given as follows:*

- *the functor T is defined on objects X by TX and on morphisms $f : X \to Y$ by the unique homomorphism between (TX, τ_X) and (TY, τ_Y) extending $\eta_Y \cdot f$;*
- *the components of η are given by η_X for every X;*
- *the components of μ are the unique homomorphisms between (TTX, τ_{TX}) and (TX, τ_X) extending id_{TX};*
- *$\bar{T}$ is given by HT;*
- *$\bar{\mu}$ is given by $H\mu$;*

- *the components of ϑ are given by τ_X for every X; and*
- *the components of κ are given by $\tau_X \cdot H\eta_X$ for every X.*

Remark 2.48. From the proof of Theorem 2.47 we have that the τ_X form a natural transformation τ. From the definition of μ we obtain $\mu_X \cdot \tau_{TX} = \tau_X \cdot H\mu_X$ for every object X; together this means $\mu \cdot \tau T = \tau \cdot H\mu$.

Lemma 2.49 ([MM06], Corollary 3.17)**.** *It holds $\tau = \mu \cdot \kappa T$.*

Proof. See the proof of Lemma 2.24 and replace ϕ by τ and F by T. Instead of Remark 2.23 use Remark 2.48. □

The following definition allows us to state the precondition of Theorem 2.47 in a shorter way.

Definition 2.50. A functor H is called *iteratable* if it has the following three equivalent properties:

- for every object X the free H-CIA on X exists;
- for every object X the final $(H(-) + X)$-coalgebra exists;
- the free CIM on H exists.

To see that the three conditions from Definition 2.50 are indeed equivalent, use Corollary 2.35, Theorem 2.47 and see [Mil05], Theorem 6.1.

Example 2.51 (see [AAMV03])**.** Every accessible set functor is iteratable.

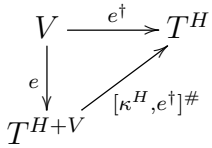
For further examples see [AAMV03].

Recursive Program Schemes

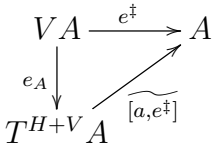
Definition 2.52 ([MM06])**.** Let H and V be endofunctors on a category with coproducts such that H and $H + V$ are iteratable. A *recursive program scheme* (or *RPS*, for short) (with givens in H) is a natural transformation $e : V \to T^{H+V}$. It is called *guarded* if it factors as

$$e \equiv (V \xrightarrow{e'} HT^{H+V} \xrightarrow{\mathsf{in}\mathsf{l}T^{H+V}} (H+V)T^{H+V} \xrightarrow{\tau^{H+V}} T^{H+V})$$

for some natural transformation $e' : V \to HT^{H+V}$. An *uninterpreted solution* of e with givens in H is an ideal natural transformation $e^\dagger : V \to T^H$ such that the diagram



commutes. An *interpreted solution* of e with givens in H in an H-CIA (A, a) is a morphism $e^\ddagger : VA \to A$ such that the diagram



commutes, where $\widetilde{[a, e^\ddagger]}$ is the unique Eilenberg-Moore algebra with $\widetilde{[a, e^\ddagger]} \cdot \kappa_A^{H+V} = [a, e^\ddagger]$ (see Definition 3.2, Lemma 3.5 and Notation 3.6 below).

Since we shall encounter several formats of recursive program schemes throughout the thesis, we refer to the RPS's of Definition 2.52 as "ordinary RPS's" when convenient.

Theorem 2.53 ([MM06])**.** *Every guarded RPS has a unique uninterpreted solution.*

Theorem 2.54 ([MM06])**.** *Every guarded RPS with givens in H has a unique interpreted solution in every H-CIA.*

2.4 Related Work

There are many introductory texts on category theory of which we shall mention only a few. The standard textbook is MacLane's [Mac98], the recent book [Sim11] by Simmons provides an easy introduction to the most basic concepts, and [AHS09] by Adámek, Herrlich and Strecker is a more faceted book with a freely available online edition. For shorter introductions see for example the lecture notes of Turi [Tur01] or van Oosten [vO95]. The notions from category theory introduced in Section 2.1 are standard; for literature on distributive laws cf. Section 1.3.

The standard reference for coalgebra is Rutten's paper [Rut00]; a further introduction is [Gum99] by Gumm. Texts which cover both algebras and coalgebras for a functor are the tutorial [JR97] by Jacobs and Rutten and the paper [Adá05] by Adámek. The latter also has a short section on complete iterativity.

Completely iterative algebras and completely iterative monads were investigated by Milius [Mil05]; they go back to the completely iterative theories of Elgot, Bloom and Tindell [EBT78]. Complete Elgot algebras were introduced and studied by Adámek, Milius and Velebil [AMV06a]. The notion of an iteratable functor was studied by Aczel, Adámek, Milius and Velebil [AAMV03]. Recursive program schemes as introduced at the end of Section 2.3 are the

topic of the paper [MM06] by Milius and Moss with precursor [GLM03] by De Marchi, Ghani and Lüth. They go back to the classical notion of a recursive program scheme, see e. g. Guessarian [Gue81].

Part I

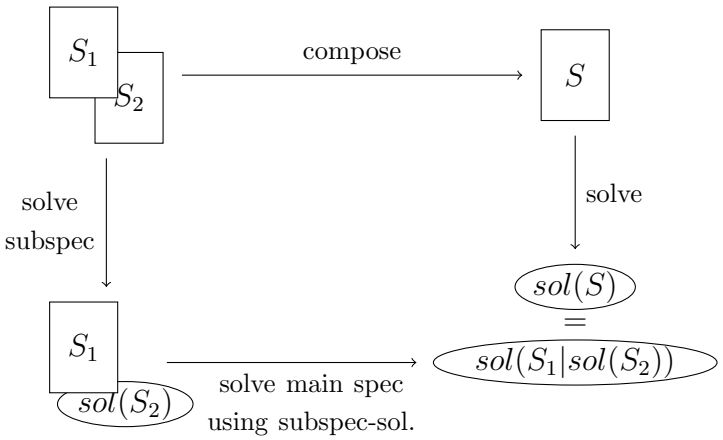
Compositionality Results for Recursive Specifications

We now enter the first main part of our work. It is concerned with *compositionality* of formats of recursive specifications as we explain next. Recall the terminology introduced in Section 1.1.

Compositionality of Formats of Recursive Specifications

Compositionality is a highly desirable property, especially useful for large specifications: it allows to compose smaller specifications into larger ones and thus reduces the complexity of specifications given in a compositional way. This is helpful for avoiding mistakes in specifications and for understanding them easier. Moreover, compositional specifications have modular solutions which can be taken step by step, and they enjoy much better maintainability.

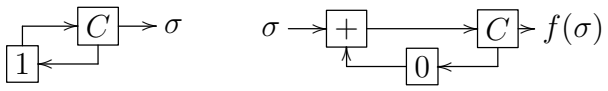
Let us be more precise what we mean by "compositionality". Suppose we are given two arbitrary recursive specifications S_1 and S_2 of some specification format where S_1 may depend on the solution of S_2. We call S_1 the main specification and S_2 its subspecification. Furthermore assume that the specification format allows to compose two specifications, and let S be the composite of S_1 and S_2. Then the specification format has the property of being *compositional* if taking the solution $sol(S)$ of S gives the same result as first solving S_2 to $sol(S_2)$ and then solving S_1 to $sol(S_1|sol(S_2))$ using $sol(S_2)$, i. e. it is compositional if $sol(S) = sol(S_1|sol(S_2))$ holds. We illustrate this in the following picture:



An Example Specification Using Stream Circuits

Let us come back to the specification of streams, i. e. elements from $\mathbb{R}^\omega$. Rutten [Rut05b] showed that streams and also functions on streams can be specified by *stream circuits*. First, we need some basic circuits: an adder $\rightarrow\boxed{+}\leftarrow$ which performs componentwise addition of streams as in (1.4), a register $\rightarrow\boxed{r}\leftarrow$ which prefixes a stream with the real number r as in (1.5), and a

copier $-\boxed{C}<$ which simply outputs two copies of an incoming stream. Stream circuits are then constructed from basic circuits by plugging wires together; we also add arrow tips to indicate the direction of data flow. These circuits are recursive specifications since they may have loops. For example, the following picture shows two stream circuits:

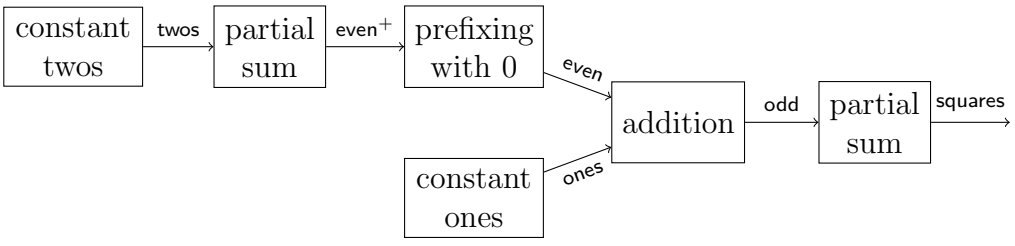


(I.1)

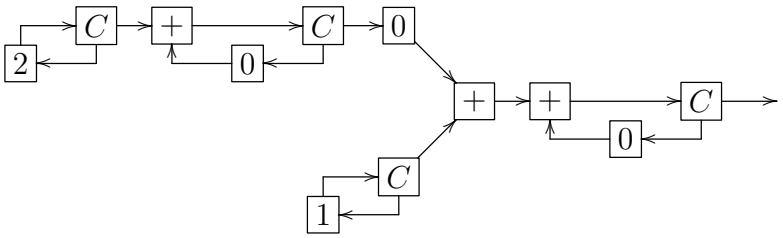
The left-hand circuit has no input and one output: it defines a stream σ. The right-hand circuit has one input and one output: it defines a unary stream function f. To understand how the stream circuits work, one needs to know that the data flow is clocked and that registers are the only memory elements which output the next number of the incoming stream one clock delayed; initially they output their initial value r. Thus the left-hand circuit in (I.1) outputs the stream $\mathsf{ones} = [1, 1, 1, \dots]$, and the right-hand one computes the partial-sum function $f([\sigma_0, \sigma_1, \sigma_2, \dots]) = [\sigma_0, \sigma_0 + \sigma_1, \sigma_0 + \sigma_1 + \sigma_2, \dots]$.

Now suppose we want to define the stream $\mathsf{squares} = [1, 4, 9, 16, \dots]$ of squares of all positive natural numbers by giving a stream circuit. It is not so easy to come up immediately with the right circuit. However, as we shall see in Remark 5.40 below, we have a compositionality principle for specifications by stream circuits: if we plug outputs of one (sub-)circuit to the inputs of a second (main) circuit, the resulting circuit will behave as the composite of the behaviors of its components. Thus it suffices to decompose our problem and give (simpler) circuits for all of its components, then our compositionality principle tells us that the composed circuit defines the desired stream.

We start the decomposition with the observation that the components of the stream $\mathsf{squares}$ are the partial sums of the stream $\mathsf{odd} = [1, 3, 5, 7, \dots]$ of odd natural numbers. And odd in turn is the componentwise addition of the streams $\mathsf{even} = [0, 2, 4, 6, \dots]$ and ones. We continue decomposing even which is the stream $\mathsf{even}^+ = [2, 4, 6, 8, \dots]$ prefixed by 0. Finally, the components of even^+ are the partial sums of the stream $\mathsf{twos} = [2, 2, 2, 2, \dots]$. Our decomposed system looks as follows:



Now it is easy to find stream circuits for all the components: a circuit which computes the constant stream ones is given by the left-hand picture in (I.1), and modifying the initial value of the register to 2 in this circuit yields a circuit which computes the constant stream twos. A circuit for the partial-sum function is given by the right-hand picture in (I.1), and for prefixing with 0 and addition we use the basic register and adder circuits. The resulting circuit is shown in the following picture:



(I.2)

It indeed computes the stream squares as we shall see in Remark 5.40.

General Setting and Techniques

An important property of the set $\mathbb{R}^\omega$ of streams is that it forms a *final coalgebra* for the endofunctor $HX = \mathbb{R} \times X$ on the category Set of sets and functions, see [MA86, Rut00]. Further examples of final coalgebras include

- the set of all infinite binary trees with node labels in $\mathbb{R}$ for the endofunctor $HX = X \times \mathbb{R} \times X$ on Set (see [MA86, SR10]),
- the set of all CCS agents or CCS processes over the action set A modulo strong bisimilarity for the endofunctor $HX = \mathcal{P}_\kappa(A \times X)$ on Set (cf. [Acz88] and see Section 4.2.3 below),
- the set $\mathcal{P}(A^*)$ of all formal languages over the alphabet A for the endofunctor $HX = X^A \times 2$ on Set (see [MA86]), and
- the class of all non-well-founded sets for the endofunctor $HX = \mathcal{P}X$ on the category Class of classes and functions (see [Acz88]).

In Part I of this thesis we shall concentrate on recursive specifications on final coalgebras. We provide several formats of recursive equations and recursive program schemes that can be used in a compositional way, and give examples of composed specifications on the five final coalgebras mentioned above.

An important tool for our work are *abstract GSOS rules* [TP97, LPW04, Bar03]. They serve to identify operations which can be used in recursive specifications (like prefixing and stream addition in the stream circuits above).

Abstract GSOS rules can be viewed as an existing format for recursive specifications of operations on final coalgebras. For example, consider again streams: here the functor $HX = \mathbb{R} \times X$ corresponds to the signature of a basic operation symbol $r.-$ (which is interpreted as prefixing a stream with r in the final coalgebra for H). To define the operation of componentwise stream addition, we provide the binary operation symbol $+$ and give the abstract GSOS rule $r.x + s.y = (r + s).(x + y)$ where the left-hand operands and also the right-hand result are structured by the basic operation symbol. Now the desired operation of componentwise stream addition is the unique interpretation of $+$ in the final coalgebra of streams.

We finally remark that abstract GSOS rules correspond to a certain kind of *distributive laws* [Bec69] from category theory, see [TP97, LPW00]. For most of our work in Part I and all our examples there, abstract GSOS rules will be sufficient. However, our formats of recursive equations can all be generalized to work with operations defined by a wider class of distributive laws as we shall show. From our formats of recursive program schemes, only some can be generalized.

Overview of Part I

We give a brief overview of this part of the thesis which consists of the Chapters 3 to 5. In Chapter 3 we recall how operations on final coalgebras can be defined by abstract GSOS rules or, more generally, by distributive laws. It serves as a foundation for the following two chapters which contain the actual compositionality results: Chapter 4 is concerned with several formats of recursive equations, and in Chapter 5 we investigate formats of recursive program schemes.

Chapter 3

Algebraic Operations from Distributive Laws

The different formats of recursive equations and recursive program schemes we shall discuss in Chapters 4 and 5 have one thing in common: all operations that may be used in the recursive specifications must be definable by abstract GSOS rules or, more generally, by distributive laws. In the present chapter, we explain what an abstract GSOS rule is and how it (or a distributive law) defines operations on a final coalgebra. None of the material in this chapter is new, but we review it since it forms the foundation of the following chapters. Along the way, the reader becomes familiar with some assumptions and notations that are kept throughout Part I of this thesis.

The present chapter is structured as follows: in Section 3.1 we recall the notions of an algebra for a pointed functor or monad and of a coalgebra for a copointed functor. In Section 3.2 we compile several variants of distributive laws defining operations on final coalgebras. The functors involved may have an additional structure, so that the distributive laws are natural transformations satisfying some laws. Section 3.3 explains how in the special (but important) case of a distributive law of a free monad over a cofree copointed functor the distributive law corresponds to a (plain) natural transformation called "abstract GSOS rule", and also how the defined operations arise directly from the latter.

We shall assume the following for the whole chapter (and moreover for the whole first part of this thesis).

Assumption 3.1. We assume that our base category (which is never mentioned explicitly) has binary products, and whenever we write a functor K, we assume that for every object X the free K-algebra on X exists.

These assumptions are relatively weak, for example Set has binary prod-

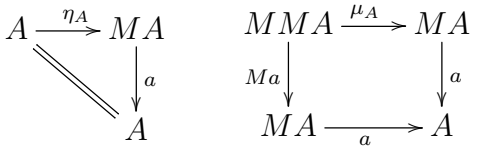
ucts and every accessible endofunctor on Set has all free algebras, see e. g. [AT90]. Notice that by Theorem 2.22 the free monad on K exists.

3.1 Algebras for Monads and (Co-)Algebras for (Co-)Pointed Functors

Algebras and coalgebras for (plain) functors have been introduced in Section 2.2. Here we consider algebras and coalgebras for functors with an additional structure where compatibility with the additional structure is required. We take a look at

- algebras for a monad, especially for a free monad and for a free CIM;
- algebras for a pointed functor; and
- coalgebras for a copointed functor, especially for a cofree copointed functor.

Definition 3.2. An *Eilenberg-Moore algebra for the monad* (M, η, μ) is an M-algebra (A, a) that makes the following diagrams



commute, which are called the *unit law* and the *multiplication law* for (A, a), respectively.

The Eilenberg-Moore algebras for a monad M form a full subcategory (M, η, μ)-Alg of M-Alg.

We shall be interested in Eilenberg-Moore algebras for free monads and free CIMs.

Lemma 3.3 ([Bar70])**.** *Let* (F, η, μ) *be the free monad on* K*. Then* (F, η, μ)-Alg *is isomorphic to* K-Alg*.*

Notation 3.4. We denote the Eilenberg-Moore algebra for a free monad F corresponding to an algebra (A, a) by $(A, \widehat{a})$.

From the proof of Lemma 3.3 it follows that $\widehat{a} \cdot \kappa_A = a$, where $\kappa : K \to F$ is the universal natural transformation of the free monad F. It also follows that $\widehat{a}$ is the unique K-algebra homomorphism from the free algebra on A to the algebra (A, a) such that $\widehat{a} \cdot \eta_A = \mathrm{id}_A$.

Lemma 3.5. *Let (T, η, μ) be the free CIM on K. Then there is a full subcategory of (T, η, μ)-*Alg *isomorphic to K-*CIA*.*

Proof (variation of the proof of Lemma 3.3 from [Bar70]). Let (T, η, μ) be the free CIM on K with universal natural transformation $\kappa : K \to T$.

Given a K-CIA (A, a), we obtain the T-algebra (A, b) where $b : TA \to A$ is the unique K-algebra homomorphism from the free CIA on A to the CIA (A, a) such that $b \cdot \eta_A = \mathrm{id}_A$. The latter is the unit law of an Eilenberg-Moore algebra already, and the multiplication law follows since both sides of $b \cdot \mu_A = b \cdot Tb$ are the unique K-algebra homomorphism h from the free CIA on TA to the CIA (A, a) such that $h \cdot \eta_{TA} = b$.

Conversely, from any Eilenberg-Moore algebra (A, b) for T we obtain the K-algebra (A, a) where $a = b \cdot \kappa_A$.

If we restrict to those Eilenberg-Moore algebras for which the second construction actually gives a K-CIA, it is not difficult to prove that both constructions are inverses of each other and that the homomorphisms between two K-CIAs are precisely the homomorphisms between the corresponding Eilenberg-Moore algebras for T. □

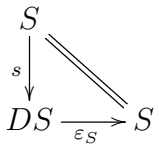
Notation 3.6. We denote the Eilenberg-Moore algebra for a free CIM T corresponding to a CIA (A, a) by $(A, \widetilde{a})$.

Now we turn to algebras for pointed functors—this notion is a restriction of Definition 3.2:

Definition 3.7. An *algebra for the pointed functor* (M, η) is an M-algebra (A, a) satisfying the unit law from Definition 3.2.

We also need to consider the dual of Definition 3.7:

Definition 3.8. A *coalgebra for the copointed functor* (D, ε) is a D-coalgebra (S, s) that makes the following diagram



commute, which is called the *counit law* for (S, s).

The coalgebras for a copointed functor D form a full subcategory (D, ε)-Coalg of D-Coalg.

Recall from Example 2.12 that in our setting of a category with binary products, the cofree copointed functor on H is $(H \times \mathrm{Id}, \mathsf{outr})$ together with outl.

Lemma 3.9 (e. g. [LPW00], Proposition 4.2)**.** *Let* $(H \times \mathrm{Id}, \mathsf{outr})$ *be the cofree copointed functor on* H *with (co)universal natural transformation* $\mathsf{outl} : (H \times \mathrm{Id}) \to H$*. Then* $(H \times \mathrm{Id}, \mathsf{outr})$-$\mathsf{Coalg}$ *is isomorphic to* H-Coalg.

3.2 Operations Obtained from Distributive Laws

In this section we see how distributive laws induce operations on a final coalgebra. Actually, they induce algebras for some functor which is possibly pointed or even a monad, but since in the case of algebras for a polynomial set functor for a finitary signature these are classical algebraic operations, we like to use the term "operations" also for different functors and categories.

From this section on and for the rest of Part I of this thesis, we make the following

Assumption 3.10. We write H for a functor on our base category which has a final coalgebra (C, c).

Now recall the types of distributive laws from Definitions 2.13 and 2.14.

Construction 3.11. Let $\lambda : KH \to HK$ be a distributive law of the endofunctors K and H. By finality of (C, c) there is a unique K-algebra (C, b) such that the diagram

$$\begin{array}{ccccc} KC & \xrightarrow{Kc} & KHC & \xrightarrow{\lambda_C} & HKC \\ {\scriptstyle b}\downarrow & & & & \downarrow{\scriptstyle Hb} \\ C & & \xrightarrow[c]{} & & HC \end{array}$$

commutes. We call (C, b) the *λ-interpretation* (in C).

If $K = M$ is a pointed functor or a monad and λ a distributive law of the pointed functor or monad M over H, we can perform the same construction, of course. But in these cases, we have additional information about the λ-interpretation:

Lemma 3.12 ([Bar04], Corollaries 3.4.12 and 3.4.19)**.** *1. If* $\lambda : MH \to HM$ *is a distributive law of a pointed functor* M *over the functor* H*, then the* λ*-interpretation* $b : MC \to C$ *is an algebra for the pointed functor.*

2. If $\lambda : MH \to HM$ *is a distributive law of a monad* M *over the functor* H*, then the* λ*-interpretation* $b : MC \to C$ *is an Eilenberg-Moore algebra.*

We can consider Construction 3.11 also for a distributive law λ of the functor K over the cofree copointed functor $(H \times \mathrm{Id}, \mathsf{outr})$. From Lemma 3.9 it follows that $\langle c, id_C \rangle : C \to HC \times C$ is the final coalgebra for $(H \times \mathrm{Id}, \mathsf{outr})$ (i. e. it is final among the coalgebras for the copointed functor). And we easily check that the coalgebra $\lambda_C \cdot K\langle c, id_C \rangle$ becomes a coalgebra for the copointed functor $H \times \mathrm{Id}$ since $\mathsf{outr}_{KC} \cdot \lambda_C \cdot K\langle c, id \rangle = K\mathsf{outr}_C \cdot K\langle c, id_C \rangle = \mathrm{id}_{KC}$ by the counit law for λ.

Construction 3.13. Let $\lambda : K(H \times \mathrm{Id}) \to (H \times \mathrm{Id})K$ be a distributive law of the functor K over the cofree copointed functor $(H \times \mathrm{Id}, \mathsf{outr})$, and let $(C, \langle c, \mathrm{id}_C \rangle)$ be the final $(H \times \mathrm{Id}, \mathsf{outr})$-coalgebra. By finality there is a unique K-algebra (C, b) such that the diagram

$$\begin{array}{ccccc} KC & \xrightarrow{K\langle c, \mathrm{id}_C\rangle} & K(HC \times C) & \xrightarrow{\lambda_C} & (H \times \mathrm{Id})KC \\ {\scriptstyle b}\downarrow & & & & \downarrow{\scriptstyle (H \times \mathrm{Id})b} \\ C & & \xrightarrow[\langle c, \mathrm{id}_C\rangle]{} & & HC \times C \end{array}$$

commutes. Again we call (C, b) the λ-*interpretation* (in C).

Again we can perform the same construction if $K = M$ is a pointed functor or a monad, and Lemma 3.12 remains valid for the respective distributive laws of M over $H \times \mathrm{Id}$ (cf. [Bar04], Corollary 3.5.1).

3.3 Operations Obtained from Abstract GSOS Rules

In this section, we consider distributive laws $\lambda : F(H \times \mathrm{Id}) \to (H \times \mathrm{Id})F$ of the free monad F on K over the cofree copointed functor $H \times \mathrm{Id}$. These distributive laws turn out to be equivalent to mere natural transformations ℓ called "abstract GSOS[1] rules". The λ-interpretation induced by the former is an Eilenberg-Moore algebra for the monad F, as we have seen in Section 3.2. There is also an ℓ-interpretation induced by the latter which is a K-algebra, and it turns out that the λ-interpretation and the ℓ-interpretation correspond to each other according to Lemma 3.3.

[1]SOS [AFV01, Plo04] stands for "structured operational semantics" or "structural operational semantics". It means that a semantics (e. g. for a programming language or a process calculus) is given by rules. GSOS [BIM95] stands for "guarded recursion SOS" and is a concrete SOS rule format.

Lemma 3.14 ([LPW04], Theorem 10)**.** *Distributive laws $\lambda : F(H \times \mathrm{Id}) \to (H \times \mathrm{Id})F$ of the free monad F on K over the cofree copointed functor $H \times \mathrm{Id}$ are in one-to-one correspondence with natural transformations $\ell : K(H \times \mathrm{Id}) \to HF$.*

Definition 3.15. An *abstract GSOS rule* is a natural transformation

$$\ell : K(H \times \mathrm{Id}) \to HF\,.$$

Remark 3.16. The name "abstract GSOS" is due to Turi and Plotkin [TP97] and is motivated by the fact that in case H is the labeled transition systems functor $\mathcal{P}_{\mathrm{f}}(A \times -)$ and K is a polynomial functor for a finitary signature, natural transformations ℓ as in Definition 3.15 correspond precisely to the GSOS rules of Bloom, Istrail and Meyer [BIM95]. Natural transformations as in Definition 3.15 have been named "abstract operational rules" by Lenisa, Power and Watanabe [LPW00, LPW04]; later Bartels [Bar04] used the name "abstract GSOS rules".

Theorem 3.17 ([Bar04, TP97])**.** *Let $\ell : K(H \times \mathrm{Id}) \to HF$ be an abstract GSOS rule. There is a unique K-algebra $b : KC \to C$ such that the diagram*

$$\begin{array}{ccccc} KC & \xrightarrow{K\langle c, \mathrm{id}_C\rangle} & K(HC \times C) & \xrightarrow{\ell_C} & HFC \\ {\scriptstyle b}\downarrow & & & & \downarrow{\scriptstyle H\widehat{b}} \\ C & & \xrightarrow[c]{} & & HC \end{array} \tag{3.1}$$

commutes. We call (C, b) the ℓ-interpretation *(in C).*

Remark 3.18. In concrete examples, one often does not need the full generality of an abstract GSOS rule. Instead, one gives a natural transformation of simpler type that gives rise to an abstract GSOS rule in a canonical way:

- a natural transformation $\ell' : KH \to HF$ is extended to an abstract GSOS rule via $\ell = (K(H \times \mathrm{Id}) \xrightarrow{K\mathsf{outl}} KH \xrightarrow{\ell'} HF)$;
- a natural transformation $\ell' : K(H \times \mathrm{Id}) \to HK$ is extended to an abstract GSOS rule via $\ell = (K(H \times \mathrm{Id}) \xrightarrow{\ell'} HK \xrightarrow{H\kappa} HF)$;
- combining the two previous items, a natural transformation $\ell' : KH \to HK$ is extended to an abstract GSOS rule via

$$\ell = (K(H \times \mathrm{Id}) \xrightarrow{K\mathsf{outl}} KH \xrightarrow{\ell'} HK \xrightarrow{H\kappa} HF)\,.$$

Lemma 3.19 ([Bar04], Lemma 3.5.2)**.** *Let ℓ be an abstract GSOS rule and let λ be the corresponding distributive law (according to Lemma 3.14). Then the Eilenberg-Moore algebra $\widehat{b} : FC \to C$ corresponding to the ℓ-interpretation $b : KC \to C$ is the λ-interpretation, i. e. the following diagram commutes:*

$$\begin{array}{ccccc} FC & \xrightarrow{F\langle c,\mathrm{id}_C\rangle} & F(HC \times C) & \xrightarrow{\lambda_C} & (H \times \mathrm{Id})FC \\ {\scriptstyle \widehat{b}}\downarrow & & & & \downarrow{\scriptstyle (H\times \mathrm{Id})\widehat{b}} \\ C & & \xrightarrow[\langle c,\mathrm{id}_C\rangle]{} & & HC \times C \end{array} \tag{3.2}$$

We remark that although this lemma was stated only for the category Set by Bartels, it holds more generally in our setting, i. e. under Assumptions 3.1 and 3.10.

3.4 Related Work

As mentioned in the beginning of the chapter, all of its contents are known. Basically the whole Chapter 3 and in particular the constructions from Sections 3.2 and 3.3 can be found in Bartels' work [Bar03, Bar04].

Lemma 3.3 goes back to Barr [Bar70], and Lemmata 3.5 and 3.9 are variations thereof. The observation that distributive laws of monads over comonads capture GSOS rules is due to Turi and Plotkin [TP97]. Subsequently, Lenisa, Power and Watanabe [LPW00, LPW04] suggested to work with distributive laws of a monad over a copointed functor since those correspond precisely to the GSOS format.

Historically, Plotkin [Plo81] invented structural operational semantics (SOS). Since then many formats for SOS rules have been developed. An introduction to SOS and several types of distributive laws has been published by Klin [Kli11]. In contrast to his and many other publications we do not stress the more general bialgebraic approach in our thesis since we exclusively work with final coalgebras.

One of the more popular SOS rule formats is the GSOS format which stems from Bloom, Istrail and Meyer [BIM95]. The name "abstract GSOS" we use in Definition 3.15 goes back to Turi and Plotkin [TP97] who use it more informally for the generalization of the GSOS format obtained by abstracting away from labeled transition systems to arbitrary coalgebra/behavior functors.

Chapter 4

Compositionality of Recursive Equations

In this chapter we introduce several formats of systems of recursive equations. Each format comes with a notion of a solution of a system of recursive equations, and we prove that these solutions always exist uniquely. We start with basic formats and proceed to more comprehensive ones; overviews over the formats and their relationships are given at the end of Sections 4.1 and 4.4.

The first formats we will encounter have no compositionality property. This is due to the fact that they do not allow *parameters* in the recursive equations. We add parameters for the following formats; then it follows from the known compositionality property of CIAs that systems of recursive equations with parameters can be composed and that these formats are compositional (cf. the introduction to Part I).

We give an overview of the present chapter. Section 4.1 contains the main results: several formats of systems of recursive equations, uniqueness results for solutions, and compositionality results. All operations used in the formats introduced in this section arise from abstract GSOS rules. In Section 4.2 we give examples by considering systems of recursive equations on five different final coalgebras. We also recover some known formats for recursive specifications. Section 4.3 is an excursion devoted to "finite" systems of recursive equations. These are the systems of interest in practical applications and are more "colorful" in the sense that—unlike the "possibly infinite" systems of recursive equations we considered before—their expressiveness varies depending on which operations are used. We prove that the solutions of finite systems of recursive equations with certain combinations of operations give rise to know subsets of three final coalgebras. Section 4.4 reviews the results of Section 4.1 for systems of recursive equations which use operations that do not arise from abstract GSOS rules but from certain distributive laws;

thus it widens the scope of the results from Section 4.1.

For Chapter 4 (and moreover for the rest of Part I) we make the following

Assumption 4.1. In addition to Assumptions 3.1 and 3.10, we assume our base category to have also binary coproducts.

As in Section 3.3, we write F for the free monad on an endofunctor K throughout the whole chapter, and we denote by $\ell : K(H \times \mathrm{Id}) \to HF$ an abstract GSOS rule.

4.1 Recursive Equations with Parameters

Recall from Theorem 2.33 that the inverse of the final H-coalgebra (C, c) is an initial H-CIA (C, c^{-1}). Thus, by giving flat equation morphisms $X \to HX + C$, we can define for example a unique element from C for each variable from X if we work in Set. However, the definition format given by these equation morphisms is not very powerful. For example, recall from the introduction of Part I that the final coalgebra of the set functor $HX = \mathbb{R} \times X$ is the set $\mathbb{R}^\omega$ of all streams. If we want to define the stream $[0, 1, 1, 2, 3, 5, \dots]$ of Fibonacci numbers (as the solution of the variable $x_0 \in X$) we can write an infinite system of recursive equations

$$\begin{aligned} x_0 &= 0.x_1 \\ x_1 &= 1.x_2 \\ x_2 &= 1.x_3 \\ x_3 &= 2.x_4 \\ x_4 &= 3.x_5 \\ x_5 &= 5.x_6 \\ &\vdots \end{aligned} \tag{4.1}$$

without using parameters from C, or a finite system of recursive equations

$$\begin{aligned} x_0 &= 0.x_1 \\ x_1 &= 1.x_2 \\ x_2 &= 1.x_3 \\ x_3 &= 2.x_4 \\ x_4 &= 3.x_5 \\ x_5 &= 5.x_6 \\ &\vdots \\ x_n &= \sigma \end{aligned} \tag{4.2}$$

using the constant stream $\sigma = \mathsf{tl}^n([0,1,1,2,3,5,\dots])$ as a parameter from C. In both cases we obviously encode the Fibonacci numbers directly into the system of recursive equations. But as we know from the introduction (cf. (1.3) in Section 1.1), the Fibonacci numbers can be defined (as the solution of the variable $x \in X$) by the very simple system of recursive equations

$$\begin{aligned} x &= 0.y \\ y &= 1.(y + x) \end{aligned} \tag{4.3}$$

where we additionally use the operation $+$ of componentwise stream addition. It is the aim of this section to provide powerful formats for recursive definitions on carriers C of final coalgebras. Moreover, we look for formats that are compositional, i. e. data defined by such a format should be reusable in further definitions and the use in a further definition should give the same solution as the composite of the recursive definitions, cf. the introduction to Part I.

As a first step, in Section 4.1.1 we show how to solve recursive definitions without parameters uniquely that involve additional operations as stream addition above. These operations will be given by the abstract GSOS rules from Section 3.3. On top of that, in Section 4.1.2 we also allow the use of parameters from C in recursive definitions. Section 4.1.3 shows that also infinite terms on the right-hand sides of recursive definitions can be allowed.

4.1.1 Parameter-Free Recursive Equations

Definition 4.2. An *ℓ-equation* is an HF-coalgebra; that is, a morphism of the form $e : X \to HFX$. A *solution* of e in the final H-coalgebra C is a morphism $e^\dagger : X \to C$ such that the diagram below commutes:

$$\begin{array}{ccc} X & \xrightarrow{e^\dagger} & C \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle c^{-1}} \\ & & HC \\ & & \uparrow{\scriptstyle H\widehat{b}} \\ HFX & \xrightarrow[HFe^\dagger]{} & HFC \end{array} \tag{4.4}$$

Theorem 4.3 ([Bar03, Bar04])**.** *For every ℓ-equation there exists a unique solution in C.*

This result follows from Corollaries 4.3.6 and 4.3.8 and Lemma 4.3.9 in [Bar04]. The first of these results is the dual of a result obtained independently and at the same time by Uustalu, Vene and Pardo (see [UVP01],

Theorem 1), and Capretta, Uustalu and Vene [CUV06], Theorems 19 and 28 generalize this work further.

We shall need a variant of Theorem 4.3 for equation morphisms of the form $e : X \to FHFX$:

Definition 4.4. A *sandwiched ℓ-equation* is an FHF-coalgebra; that is, a morphism of the form $e : X \to FHFX$. A *solution* of e in the final H-coalgebra C is a morphism $e^\dagger : X \to C$ such that the diagram below commutes:

$$\begin{array}{ccc} X & \xrightarrow{e^\dagger} & C \\ \Big\downarrow{\scriptstyle e} & & \Big\uparrow{\scriptstyle \widehat{b}} \\ & & FC \\ & & \Big\uparrow{\scriptstyle Fc^{-1}} \\ & & FHC \\ & & \Big\uparrow{\scriptstyle FH\widehat{b}} \\ FHFX & \xrightarrow[FHFe^\dagger]{} & FHFC \end{array} \tag{4.5}$$

Theorem 4.5. *For every sandwiched ℓ-equation there exists a unique solution in C.*

Proof. Given a sandwiched ℓ-equation $e : X \to FHFX$, we form the following (ordinary) ℓ-equation:

$$\bar{e} = (\, HFX \xrightarrow{HFe} HFFHFX \xrightarrow{H\mu_{HFX}} HFHFX \,).$$

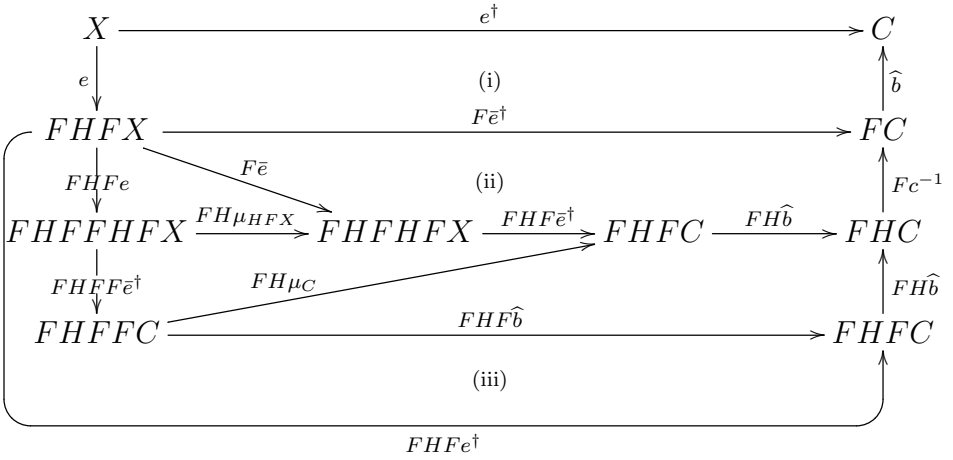
From Theorem 4.3 we know that $\bar{e}$ has a unique solution $\bar{e}^\dagger : HFX \to C$. Thus, we are finished if we can show that solutions of e and $\bar{e}$ are in one-to-one correspondence.

First, from the solution $\bar{e}^\dagger$ of $\bar{e}$ we obtain

$$e^\dagger = (\, X \xrightarrow{e} FHFX \xrightarrow{F\bar{e}^\dagger} FC \xrightarrow{\widehat{b}} C \,),$$

and we will now verify that $e^\dagger$ is a solution of e. To this end, consider the

diagram below:

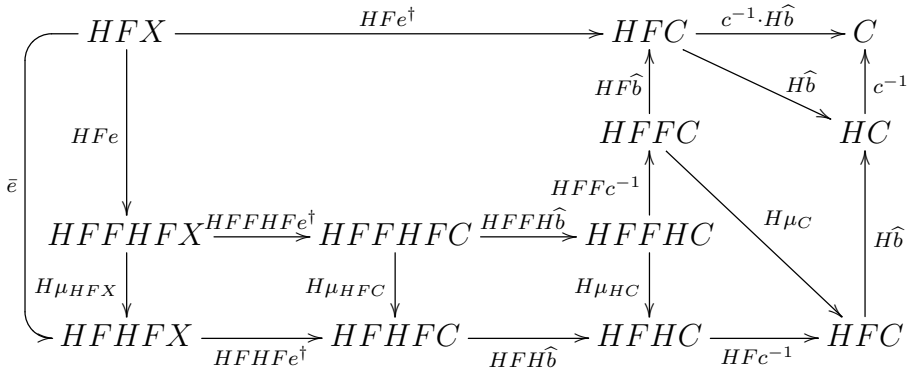


All its inner parts commute: part (i) and (iii) commute by the definition of $e^\dagger$, for part (ii) use that $\bar{e}^\dagger$ is a solution of $\bar{e}$ (i. e. apply F to diagram (4.4) with $\bar{e}$ in lieu of e), and the remaining parts commute by the definition of $\bar{e}$, naturality of μ and the multiplication law for the Eilenberg-Moore algebra $\widehat{b} : FC \to C$. Thus, the outside commutes proving $e^\dagger$ to be a solution of e.

Second, suppose we are given any solution $e^\dagger$ of e. Then we form

$$HFX \xrightarrow{HFe^\dagger} HFC \xrightarrow{H\widehat{b}} HC \xrightarrow{c^{-1}} C\,,$$

and we now prove that this is a solution of $\bar{e}$. Indeed, in the following diagram



all inner parts commute: for the big upper left-hand square apply HF to diagram (4.5), the left-hand part is the definition of $\bar{e}$, the two lower squares and the lower right-hand triangle commute due to naturality of μ, the upper right-hand triangle is trivial, and the remaining middle right-hand part commutes by the multiplication law for the Eilenberg-Moore algebra $\widehat{b} : FC \to C$.

Thus, the outside square commutes proving $c^{-1} \cdot H\widehat{b} \cdot HFe^\dagger$ to be a solution of $\bar{e}$. Since $\bar{e}$ has a unique solution, we have

$$c^{-1} \cdot H\widehat{b} \cdot HFe^\dagger = \bar{e}^\dagger \,.$$

Finally, the two constructions are inverse to each other: starting with the unique solution $\bar{e}^\dagger$ of $\bar{e}$, it is clear that by applying the two constructions we obtain $\bar{e}^\dagger$ again. Starting with any solution $e^\dagger$ of e, the application of the second construction results in the solution $c^{-1} \cdot H\widehat{b} \cdot HFe^\dagger = \bar{e}^\dagger$ of $\bar{e}$. The application of the first construction to that solution gives the solution $e^\dagger$ of e:

$$\widehat{b} \cdot F\bar{e}^\dagger \cdot e = \widehat{b} \cdot F(c^{-1} \cdot H\widehat{b} \cdot HFe^\dagger) \cdot e = \widehat{b} \cdot Fc^{-1} \cdot FH\widehat{b} \cdot FHFe^\dagger \cdot e = e^\dagger$$

where the last equality uses diagram (4.5). We conclude that e has a unique solution $e^\dagger$. □

Remark 4.6. Observe that already H-coalgebras $X \to HX$ constitute a basic format of systems of recursive equations, the unique solutions in C being the homomorphisms into the final H-coalgebra. The format of an ℓ-equation $X \to HFX$, for which unique solutions in C exist by Theorem 4.3, comprises (for any $\ell : K(H \times \mathrm{Id}) \to HF$) the format of an H-coalgebra: every H-coalgebra $e : X \to HX$ can be considered as an ℓ-equation by taking $H\eta_X \cdot e : X \to HFX$. One easily checks that the respective solutions coincide (i. e., the solution is preserved by the construction). Moreover, the format of a sandwiched ℓ-equation $X \to FHFX$, for which unique solutions in C exist by Theorem 4.5, comprises the format of an ℓ-equation: every ℓ-equation $e : X \to HFX$ can be considered as a sandwiched ℓ-equation by taking $\eta_{HFX} \cdot e : X \to FHFX$. Again one easily checks that the construction preserves the solution.

In the introduction to Section 4.1 we have seen in the system (4.1) of recursive equations that it is possible to define the stream of Fibonacci numbers already by giving an H-coalgebra. It is not difficult to see that for every set functor H every element of C can be defined by an H-coalgebra $e : X \to HX$: one just takes $X = C$ and considers the final H-coalgebra $e = c : C \to HC$. Its unique solution obviously is the identity on C. This indicates that the additional usage of operations obtained from a GSOS rule does not lead to more "expressive power", at least when we work in Set.

Thus for Set we conclude from the two facts that additional operations lead to a more general but equally expressive format, that we obtain additional possibilities to define the same elements from C. In fact, this is precisely the effect demonstrated in the Fibonacci numbers example above:

the additional operation of componentwise stream addition + makes the definition (4.3) of the stream of Fibonacci numbers possible. Compared to the definition (4.1) of the same stream (but without using the operation +), this new possibility has the great advantage that only two variables instead of infinitely many are needed.

4.1.2 Adding Parameters

Theorem 4.7. (Unique solutions of flat equation morphisms $X \to HFX + C$)**.** *Consider the HF-algebra*

$$k = (\, HFC \xrightarrow{H\widehat{b}} HC \xrightarrow{c^{-1}} C\,),$$

where $b : KC \to C$ is the ℓ-interpretation in C. Then (C, k) is a CIA for the functor HF.

Proof. Let $e : X \to HFX + C$ be a flat equation morphism. We must prove that there exists a unique morphism $e^\dagger : X \to C$ such that the following square commutes:

$$\begin{array}{ccc} X & \xrightarrow{\quad e^\dagger \quad} & C \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [k,C]} \\ HFX + C & \xrightarrow[HFe^\dagger + C]{} & HFC + C \end{array}$$

We start by forming the ℓ-equation

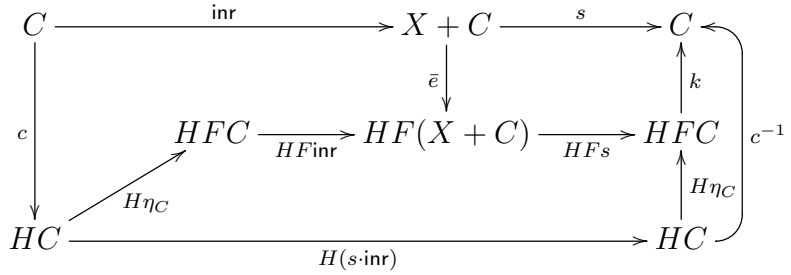
$$\bar{e} = (\, X + C \xrightarrow{[e,\mathsf{inr}]} HFX + C \xrightarrow{HFX + H\eta_C \cdot c} HFX + HFC \xrightarrow{\mathsf{can}} HF(X + C)\,).$$

By Theorem 4.3, there exists a unique morphism $s : X + C \to C$ such that the square below commutes:

$$\begin{array}{ccc} X + C & \xrightarrow{\quad s \quad} & C \\ \downarrow & & \uparrow{\scriptstyle c^{-1}} \\ {\scriptstyle \bar{e}}\downarrow & & HC \\ & & \uparrow{\scriptstyle H\widehat{b}} \\ HF(X + C) & \xrightarrow[HFs]{} & HFC \end{array} \tag{4.6}$$

We will now prove that the morphism $e^\dagger = s \cdot \mathsf{inl} : X \to C$ is the desired unique solution of e. We begin by proving that the equation $s \cdot \mathsf{inr} = \mathrm{id}_C$

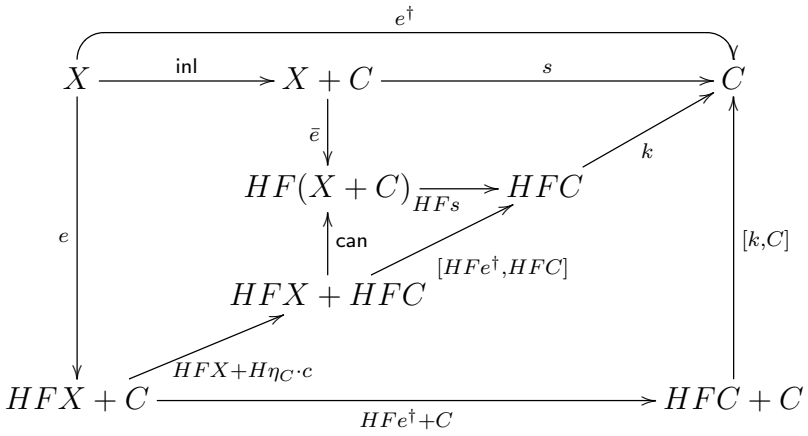
holds. Indeed, consider the diagram below:



(4.7)

This diagram commutes: the upper right-hand square is diagram (4.6) above, the upper left-hand part commutes by the definition of $\bar{e}$, the lower part commutes by the naturality of η, and the right-hand part follows from the definition of $k = c^{-1} \cdot H\widehat{b}$ and the unit law $\widehat{b} \cdot \eta_C = \mathrm{id}_C$ of the Eilenberg-Moore algebra $(C, \widehat{b})$, cf. Notation 3.4. Hence, we see that $s \cdot \mathsf{inr}$ is an H-coalgebra homomorphism from the final H-coalgebra (C, c) to itself. Thus, $s \cdot \mathsf{inr}$ must be the identity as desired.

Next we prove that $e^\dagger$ is a solution of e. To this end we verify that the following diagram commutes:



(4.8)

The upper part is the definition of $e^\dagger$, the left-hand part commutes by the definition of $\bar{e}$, the upper right-hand part is diagram (4.6), and that the inner triangle commutes follows from the definition of $e^\dagger$ and the fact that $s \cdot \mathsf{inr} = \mathrm{id}_C$. Finally, we consider the two coproduct components of the lower part separately; the left-hand component trivially commutes, and for the right-hand one we compute as follows:

$$\begin{aligned} k \cdot H\eta_C \cdot c &= c^{-1} \cdot H\widehat{b} \cdot H\eta_C \cdot c && \text{(by the definition of } k\text{)} \\ &= c^{-1} \cdot c && \text{(since } \widehat{b} \cdot \eta_C = \mathrm{id}_C\text{)} \\ &= \mathrm{id}_C\,. \end{aligned}$$

To complete our proof we show that $e^\dagger = s \cdot \mathsf{inl} : X \to C$ is the unique solution of e. So suppose that we are given any solution $e^\dagger$ of e. Now form the morphism $s = [e^\dagger, \mathrm{id}_C]$. We are finished if we show that for this morphism s the diagram (4.6) commutes. We verify the two coproduct components separately: the right-hand component is checked using diagram (4.7)—the outside of (4.7) commutes since $s \cdot \mathsf{inr} = \mathrm{id}_C$, and commutativity of the desired upper right-hand square precomposed with inr follows since all other inner parts commute as described below (4.7). The commutativity of the left-hand component is established using diagram (4.8); indeed, since the outside and all other parts of that diagram commute for our morphism s so does the desired upper right-hand square precomposed with inl. □

Theorem 4.8. (Unique solutions of flat equation morphisms $X \to FHFX + C$)**.** *Consider the FHF-algebra*

$$k' = (FHFC \xrightarrow{Fk} FC \xrightarrow{\widehat{b}} C)\,,$$

where $b : KC \to C$ is the ℓ-interpretation in C and $k = c^{-1} \cdot H\widehat{b}$. Then (C, k') is a CIA for the functor FHF.

Proof (sketch). The proof is just a slight modification of the proof of Theorem 4.7: now we are given a flat equation morphism $e : X \to FHFX + C$ and we form the morphism

$$\begin{array}{rl} \bar{e} = & X + C \xrightarrow{[e,\mathsf{inr}]} FHFX + C \\ & \qquad\qquad\qquad\quad \Big\downarrow {\scriptstyle FHFX + FH\eta_C \cdot \eta_{HC} \cdot c} \\ & \qquad FHFX + FHFC \xrightarrow{\mathsf{can}} FHF(X + C)\,. \end{array}$$

This morphism $\bar{e}$ is a sandwiched ℓ-equation and we invoke Theorem 4.5 to see that it has a unique solution s. The rest of this proof proving that $\bar{s} \cdot \mathsf{inl}$ is the unique solution of e is left to the reader since it is very close to the one of Theorem 4.7. □

Remark 4.9 (Compositionality of flat equation morphisms). Observe that with the possibility to use parameters in recursive equations, we have a mathematical explanation of *modularity* of these equations: the solution of a flat equation morphism for HF or FHF is a morphism $X \to C$, and can be used to provide parameters in a subsequent flat equation morphism, and this equation morphism again has a unique solution by Theorem 4.7 or Theorem 4.8.
Moreover, this modularity property extends to a compositionality property

which is made explicit by the following property that is true in every H-CIA (A, a), see [AMV06a]. Suppose we have two flat equation morphisms $e : X \to HX + Y$ and $f : Y \to HY + A$ where e depends on f, then we can form the composite flat equation morphism $f \blacksquare e : X + Y \to H(X + Y) + A$ (see Notation 2.36). Formalizing the above described modularity, we can also use the solution $f^\dagger : Y \to A$ of f in the CIA (A, a) as parameters in the equation morphism e by forming $f^\dagger \bullet e : X \to HX + A$ (see Notation 2.36). Then

$$(f \blacksquare e)^\dagger = (\ X + Y \xrightarrow{[(f^\dagger \bullet e)^\dagger, f^\dagger]} A\ ),$$

i. e. the solution of the composite in the CIA (A, a) can be obtained by first solving f in (A, a) to $f^\dagger$ and then solving $f^\dagger \bullet e$ in (A, a) to $(f^\dagger \bullet e)^\dagger$.

Remark 4.10. Observe that—similar as in Remark 4.6 of Section 4.1.1—the newly introduced formats of flat equation morphisms $X \to HFX + C$ (which have unique solutions in C by Theorem 4.7) and $X \to FHFX + C$ (which have unique solutions in C by Theorem 4.8) comprise other formats: given a flat equation morphism $e : X \to HX + C$ (which has a unique solution in C since $c^{-1} : HC \to C$ is a CIA by Theorem 2.33), we can form the flat equation morphism $(H\eta_X + \mathrm{id}_C) \cdot e : X \to HFX + C$ (for any $\ell : K(H \times \mathrm{Id}) \to HF$). And given a flat equation morphism $e : X \to HFX + C$, we can form the flat equation morphism $(\eta_{HFX} + \mathrm{id}_C) \cdot e : X \to FHFX + C$. Both constructions preserve solutions.

Moreover, the three formats of flat equation morphisms comprise the three formats from Remark 4.6 via left injection. More precisely, every H-coalgebra $e : X \to HX$ gives rise to a flat equation morphism $\mathsf{inl} \cdot e : X \to HX + C$, every ℓ-equation $e : X \to HFX$ gives rise to a flat equation morphism $\mathsf{inl} \cdot e : X \to HFX + C$, and every sandwiched ℓ-equation $e : X \to FHFX$ gives rise to a flat equation morphism $\mathsf{inl} \cdot e : X \to FHFX + C$. And all constructions preserve the solutions.

4.1.3 Using Infinite Terms

Recall the notion of a flat equation morphism $e : X \to HX + A$ from Definition 2.30. These equation morphisms have unique solutions in a CIA by definition. We now recall how to obtain unique solutions of more general (first-order) recursive equations than the flat ones:

Definition 4.11 ([AAMV03, Mil05])**.** Let H be iteratable. An *equation morphism* (with parameters in A) is a morphism of the form $e : X \to T^H(X + A)$. It is called *guarded* if it factors as

$$e \equiv (\ X \xrightarrow{e'} HT^H(X + A) + A \xrightarrow{[\tau^H_{X+A}, \eta^H_{X+A} \cdot \mathsf{inr}]} T^H(X + A)\ )$$

for some morphism $e' : X \to HT^H(X + A) + A$. A *solution* of an equation morphism e with parameters in A in a CIA (A, a) is a morphism $e^\dagger : X \to A$ such that the following diagram commutes:

$$\begin{array}{ccc} X & \xrightarrow{\quad e^\dagger \quad} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle \widetilde{a}} \\ T^H(X + A) & \xrightarrow[T^H[e^\dagger, \mathrm{id}_A]]{} & T^H A \end{array}$$

Theorem 4.12 ([Mil05])**.** *Let (A, a) be a CIA for an iteratable functor H. Then every guarded equation morphism with parameters in A has a unique solution in (A, a).*

Coming back to our setting in this chapter, let $\ell : K(H \times \mathrm{Id}) \to HF$ be an abstract GSOS rule, where F is the free monad on K. Assume furthermore that the composite functors HF and FHF are iteratable. By applying Theorem 4.12 to the CIAs $k : HFC \to C$ from Theorem 4.7 and $k' : FHFC \to C$ from Theorem 4.8 we get two solutions theorems for free:

Corollary 4.13. *Every guarded equation morphism $e : X \to T^{HF}(X + C)$ has a unique solution in the CIA (C, k) of Theorem 4.7.*

Corollary 4.14. *Every guarded equation morphism $e : X \to T^{FHF}(X + C)$ has a unique solution in the CIA (C, k') of Theorem 4.8.*

Remark 4.15 (Compositionality of guarded equation morphisms). The modularity and compositionality principles for flat equation morphisms from Remark 4.9 extend to guarded equation morphisms. More precisely, given guarded equation morphisms $e : X \to T^H(X + Y)$ and $f : Y \to T^H(Y + A)$ we abuse notation slightly and define

$$f^\dagger \bullet e \equiv (X \xrightarrow{\quad e \quad} T^H(X + Y) \xrightarrow{T^H(X + f^\dagger)} T^H(X + A))$$

and

$$\begin{array}{l} f \blacksquare e \equiv (\, X + Y \xrightarrow{[e, \eta^H_{X+Y} \cdot \mathsf{inr}]} T^H(X + Y) \\ \qquad\qquad\qquad\qquad\qquad\qquad \Big\downarrow {\scriptstyle T^H[T^H \mathsf{inl} \cdot \eta^H_X, T^H \mathsf{inr} \cdot f]} \\ \qquad\qquad\qquad\qquad\qquad T^H T^H(X + Y + A) \xrightarrow{\mu^H_{X+Y+A}} T^H(X + Y + A)\,)\,. \end{array}$$

Observe that both of the above equation morphisms are guarded: $f^\dagger \bullet e$ is guarded since e is, and $f \blacksquare e$ is guarded since e and f are. Then we obtain the analogous result to Remark 4.9:

$$(f \blacksquare e)^\dagger = (\, X + Y \xrightarrow{[(f^\dagger \bullet e)^\dagger, f^\dagger]} A\,)$$

Indeed, the proof is similar to that from [AMV06a] for flat equation morphisms: one proves that $[(f^\dagger \bullet e)^\dagger, f^\dagger]$ is a solution of $f \blacksquare e$, then the desired result follows from the uniqueness of solutions in a CIA.

We remark that more sophisticated compositionality results are proved in [MM09] and [Mos03], where the equation morphisms may refer to the variables of each other. However, these results, which can be found under the names "pairing identity" and "Bekič-Scott-identity", were only formulated for equation morphisms for a monad (cf. Definition 2.45) and not for (flat) equation morphisms for CIAs.

Remark 4.16. As in Sections 4.1.1 (Remark 4.6) and 4.1.2 (Remark 4.10), observe that the newly introduced formats of guarded equation morphisms $X \to T^{HF}(X + C)$ (which have unique solutions in C by Corollary 4.13) and $X \to T^{FHF}(X + C)$ (which have unique solutions in C by Corollary 4.14) comprise other formats: given a guarded equation morphism $e : X \to T^H(X + C)$ (which has a unique solution in C by Theorem 4.12 since $c^{-1} : HC \to C$ is a CIA), we can form the guarded equation morphism $(\kappa^{HF} \cdot H\eta)^\#_{X+C} \cdot e : X \to T^{HF}(X + C)$ (for any $\ell : K(H \times \mathrm{Id}) \to HF$). Here $(\kappa^{HF} \cdot H\eta)^\# : T^H \to T^{HF}$ is the unique idealized monad morphism between CIMs extending $\kappa^{HF} \cdot H\eta$, see Definition 2.46, and guardedness follows from the guardedness of e as proved in Lemma 4.17 below. Similarly, given a guarded equation morphism $e : X \to T^{HF}(X + C)$, we can form the guarded equation morphism $(\kappa^{FHF} \cdot \eta HF)^\# \cdot e : X \to T^{FHF}(X + C)$. Both constructions preserve solutions. We prove this for the first construction in Lemma 4.17 below; for the second construction the proof that guardedness and solutions are preserved is very similar.
Moreover, the three formats of guarded equation morphisms comprise the three formats of flat equation morphisms from Remark 4.10. More precisely, every flat equation morphism $e : X \to HX + C$ gives rise to a guarded equation morphism $\mathsf{can} \cdot (\kappa^H_X + \eta^H_C) \cdot e : X \to T^H(X + C)$ as proved in Lemma 4.18 below. Similarly, provided that HF and FHF are iteratable as required by Corollaries 4.13 and 4.14, flat equation morphisms $e : X \to HFX + C$ give rise to guarded equation morphisms $\mathsf{can} \cdot (\kappa^{HF}_X + \eta^{HF}_C) \cdot e : X \to T^{HF}(X + C)$ and flat equation morphisms $e : X \to FHFX + C$ give rise to guarded equation morphisms $\mathsf{can} \cdot (\kappa^{FHF}_X + \eta^{FHF}_C) \cdot e : X \to T^{FHF}(X + C)$. And each of the three constructions preserves the solutions. We prove this for the first construction in Lemma 4.18 below; for the second and third construction the proof is very similar.

Lemma 4.17. *Let $e : X \to T^H(X + C)$ be a guarded equation morphism and let $e^\dagger : X \to C$ be its unique solution in the CIA $c^{-1} : HC \to C$. Then, for any $\ell : K(H \times \mathrm{Id}) \to HF$ with ℓ-interpretation $b : FC \to C$,*

$\bar{e} = (\kappa^{HF} \cdot H\eta)^{\#}_{X+C} \cdot e : X \to T^{HF}(X+C)$ *is a guarded equation morphism, and its unique solution* $\bar{e}^\dagger : X \to C$ *in the CIA* $c^{-1} \cdot H\widehat{b} : HFC \to C$ *is* $\bar{e}^\dagger = e^\dagger$.

Proof. Since e is guarded, there is some $e' : X \to HT^H(X+C)+C$ such that $e = [\tau^H_{X+C}, \eta^H_{X+C} \cdot \mathsf{inr}] \cdot e'$. Using this equation, we see that $\bar{e}$ is also guarded since it factors through $\bar{e}' = (HF(\kappa^{HF} \cdot H\eta)^{\#}_{X+C} \cdot H\eta_{T^H(X+C)} + \mathrm{id}_C) \cdot e' : X \to HFT^{HF}(X+C) + C$:

$$\begin{aligned}
\bar{e} &= (\kappa^{HF} \cdot H\eta)^{\#}_{X+C} \cdot e \\
&= (\kappa^{HF} \cdot H\eta)^{\#}_{X+C} \cdot [\tau^H_{X+C}, \eta^H_{X+C} \cdot \mathsf{inr}] \cdot e' \\
&= [\tau^{HF}_{X+C}, \eta^{HF}_{X+C} \cdot \mathsf{inr}] \cdot (HF(\kappa^{HF} \cdot H\eta)^{\#}_{X+C} \cdot H\eta_{T^H(X+C)} + \mathrm{id}_C) \cdot e' \\
&= [\tau^{HF}_{X+C}, \eta^{HF}_{X+C} \cdot \mathsf{inr}] \cdot \bar{e}' \,.
\end{aligned}$$

The last but one equation holds since we have

$$\begin{aligned}
&(\kappa^{HF} \cdot H\eta)^{\#}_{X+C} \cdot [\tau^H_{X+C}, \eta^H_{X+C} \cdot \mathsf{inr}] \\
&= [\tau^{HF}_{X+C}, \eta^{HF}_{X+C} \cdot \mathsf{inr}] \cdot (HF(\kappa^{HF} \cdot H\eta)^{\#}_{X+C} \cdot H\eta_{T^H(X+C)} + \mathrm{id}_C) \,.
\end{aligned}$$

Indeed, consider the coproduct components separately: for the right-hand component use that $(\kappa^{HF} \cdot H\eta)^{\#}_{X+C}$ is a monad morphism to see that

$$(\kappa^{HF} \cdot H\eta)^{\#}_{X+C} \cdot \eta^H_{X+C} \cdot \mathsf{inr} = \eta^{HF}_{X+C} \cdot \mathsf{inr} \,.$$

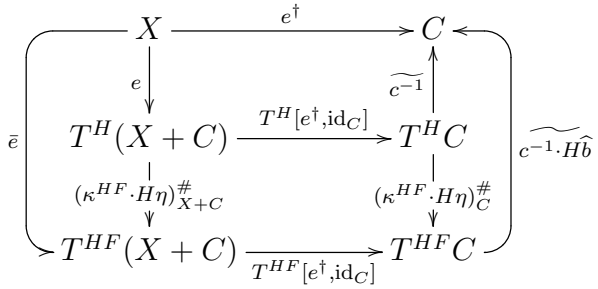
And for the left-hand component, we have

$$\begin{aligned}
&(\kappa^{HF} \cdot H\eta)^{\#}_{X+C} \cdot \tau^H_{X+C} \\
&= (\kappa^{HF} \cdot H\eta)^{\#}_{X+C} \cdot \mu^H_{X+C} \cdot \kappa^H_{T^H(X+C)} \\
&= \mu^{HF}_{X+C} \cdot T^{HF}(\kappa^{HF} \cdot H\eta)^{\#}_{X+C} \cdot (\kappa^{HF} \cdot H\eta)^{\#}_{T^H(X+C)} \cdot \kappa^H_{T^H(X+C)} \\
&= \mu^{HF}_{X+C} \cdot T^{HF}(\kappa^{HF} \cdot H\eta)^{\#}_{X+C} \cdot \kappa^{HF}_{T^H(X+C)} \cdot H\eta_{T^H(X+C)} \\
&= \mu^{HF}_{X+C} \cdot \kappa^{HF}_{T^{HF}(X+C)} \cdot HF(\kappa^{HF} \cdot H\eta)^{\#}_{X+C} \cdot H\eta_{T^H(X+C)} \\
&= \tau^{HF}_{X+C} \cdot HF(\kappa^{HF} \cdot H\eta)^{\#}_{X+C} \cdot H\eta_{T^H(X+C)} \,,
\end{aligned} \tag{4.9}$$

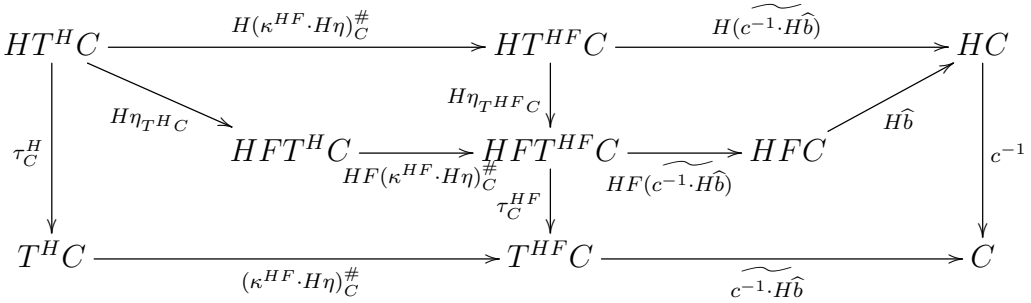
where we use Lemma 2.49 for the first and last equation and for the equations in between use that $(\kappa^{HF} \cdot H\eta)^{\#}$ is a monad morphism between the free CIMs T^H and T^{HF} which moreover extends $\kappa^{HF} \cdot H\eta$, and naturality of κ^{HF}.

To see that $\bar{e}$ has the solution $\bar{e}^\dagger = e^\dagger$ in the CIA $c^{-1} \cdot H\widehat{b} : HFC \to C$,

consider the following diagram:



The left-hand part is the definition of $\bar{e}$, the upper square commutes since $e^\dagger$ is a solution of e in the CIA $c^{-1} : HC \to C$, and the lower square commutes due to naturality of $(\kappa^{HF} \cdot H\eta)^\#$. It remains to prove commutativity of the right-hand part, then the outside commutes which prove the desired result. We do this by showing that both morphisms $\widetilde{c^{-1}}$ and $\widetilde{(c^{-1} \cdot H\widehat{b})} \cdot (\kappa^{HF} \cdot H\eta)^\#_C$ are the unique homomorphism between the free CIA $\tau^H_C : HT^H C \to T^H C$ and the CIA $c^{-1} : HC \to C$ extending id_C. For the Eilenberg-Moore algebra $\widetilde{c^{-1}}$ this is clear by its construction (see the proof of Lemma 3.5), and for $\widetilde{(c^{-1} \cdot H\widehat{b})} \cdot (\kappa^{HF} \cdot H\eta)^\#_C$ consider the following diagram:



The upper left-hand part commutes by naturality of η, for the lower left-hand part use equation (4.9) with $X + C$ replaced by C. To see that the upper right-hand part commutes, remove H and calculate

$$\widehat{b} \cdot F\widetilde{(c^{-1} \cdot H\widehat{b})} \cdot \eta_{T^{HF}C} = \widehat{b} \cdot \eta_C \cdot \widetilde{(c^{-1} \cdot H\widehat{b})} = \widetilde{c^{-1} \cdot H\widehat{b}}$$

using the naturality of η and $\widehat{b} \cdot \eta_C = \mathrm{id}_C$. The lower right-hand part commutes by the construction of the Eilenberg-Moore algebra $\widetilde{c^{-1} \cdot H\widehat{b}}$ (see again the proof of Lemma 3.5). Thus the outside commutes. Finally, we have $\widetilde{(c^{-1} \cdot H\widehat{b})} \cdot (\kappa^{HF} \cdot H\eta)^\#_C \cdot \eta^H_C = \widetilde{(c^{-1} \cdot H\widehat{b})} \cdot \eta^{HF}_C = \mathrm{id}_C$ which concludes the proof. □

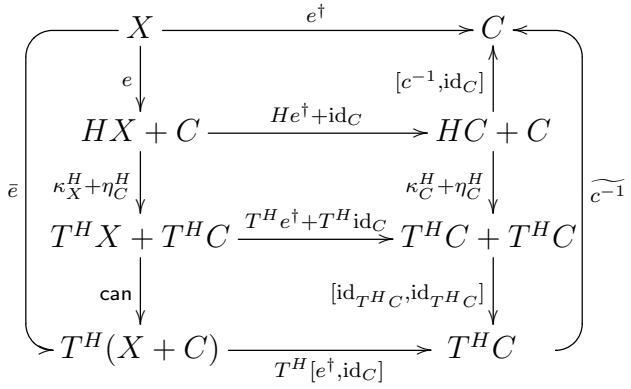
Lemma 4.18. *Let $e : X \to HX + C$ be a flat equation morphism and let $e^\dagger : X \to C$ be its unique solution in the CIA $c^{-1} : HC \to C$. Then $\bar{e} = \mathsf{can} \cdot (\kappa_X^H + \eta_C^H) \cdot e : X \to T^H(X + C)$ is a guarded equation morphism, and its unique solution $\bar{e}^\dagger : X \to C$ in the CIA c^{-1} is $\bar{e}^\dagger = e^\dagger$.*

Proof. To see that $\bar{e}$ is guarded, we calculate

$$\begin{aligned}\bar{e} &= \mathsf{can} \cdot (\kappa_X^H + \eta_C^H) \cdot e \\ &= [T^H\mathsf{inl}, T^H\mathsf{inr}] \cdot (\tau_X^H \cdot H\eta_X^H + \eta_C^H) \cdot e \\ &= [\tau_{X+C}^H, \eta_{X+C}^H \cdot \mathsf{inr}] \cdot (HT^H\mathsf{inl} \cdot H\eta_X^H + \mathrm{id}_C) \cdot e\end{aligned}$$

which exhibits $(HT^H\mathsf{inl} \cdot H\eta_X^H + \mathrm{id}_C) \cdot e$ as a factor guarding $\bar{e}$. Here the first equation is the definition of $\bar{e}$, the second one uses the definitions of can and κ^H (see Theorem 2.47), and the last one uses naturality of τ^H and η^H.

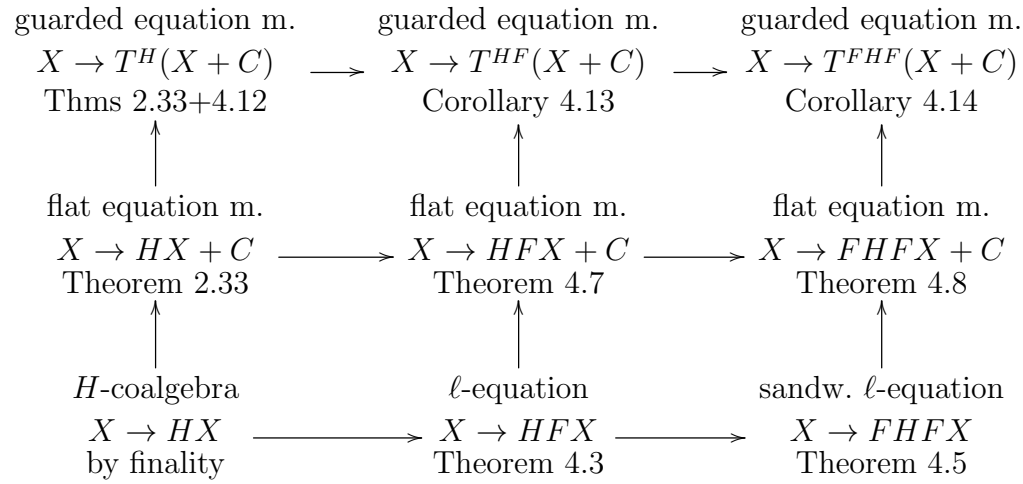
To see that $\bar{e}$ has the solution $\bar{e}^\dagger = e^\dagger$ in the CIA $c^{-1} : HC \to C$, consider the following diagram:



The left-hand part is the definition of $\bar{e}$, the upper square commutes since $e^\dagger$ is a solution of e in the CIA c^{-1}, the center square commutes by naturality of κ^H and η^H and the lower square is trivial. The two coproduct components of the remaining right-hand part commute by the equations $\widetilde{c^{-1}} \cdot \kappa_C^H = c^{-1}$ and $\widetilde{c^{-1}} \cdot \eta_C^H = \mathrm{id}_C$ for the Eilenberg-Moore algebra $\widetilde{c^{-1}} : T^HC \to C$. Thus the outside commutes which concludes the proof. □

Summarizing Remarks 4.6, 4.10 and 4.16, the format from Corollary 4.14 subsumes all the previous formats as shown in the following overview (arrows

point to more general formats):



4.2 Examples

In this section, we illustrate the uniqueness and compositionality results from Section 4.1 on five concrete final coalgebras:

- streams,
- infinite trees,
- CCS processes,
- formal languages, and
- non-well-founded sets.

The first four ones are final coalgebras for set functors, the last one is a final coalgebra for an endofunctor on the category Class of classes and functions.

Moreover, we show how our work captures Milner's solution theorem for CCS processes and the definition of context-free languages by grammars in Greibach normal form.

4.2.1 Streams

Streams have been studied in a coalgebraic setting by Rutten [Rut05a]. Here we take the functor $HX = \mathbb{R} \times X$ whose final coalgebra (C, c) is carried by the set $\mathbb{R}^\omega$ of all streams over $\mathbb{R}$ and $c = \langle \mathsf{hd}, \mathsf{tl} \rangle : \mathbb{R}^\omega \to \mathbb{R} \times \mathbb{R}^\omega$ is given by the usual head and tail functions on streams, see [MA86, Rut00].

Consider the following three operations: prefixing $r.-$, componentwise addition $+$ and scalar multiplication $r \cdot -$. We can define them by giving an abstract GSOS rule $\ell : K(H \times \mathrm{Id}) \to HF$, where we choose $HX = \mathbb{R} \times X$ to be the stream functor and $KX = \mathbb{R} \times X + X \times X + \mathbb{R} \times X$ to be the functor corresponding to the signature of the three operations, see also [Bar04], Section 3.5.1. We give ℓ componentwise for the three coproduct components of K:

- for the first component corresponding to the prefixing operation $r.x$, we define for every set X a function $\mathbb{R} \times (\mathbb{R} \times X \times X) \to \mathbb{R} \times FX$ by the assignment $(r, (s, x', x)) \mapsto (r, x)$;

- for the second component corresponding to componentwise stream addition $x+y$, we define for every set X a function $(\mathbb{R}\times X\times X)^2 \to \mathbb{R}\times FX$ by the assignment $((r, x', x), (s, y', y)) \mapsto (r + s, x' + y')$;

- for the third component corresponding to scalar multiplication $r \cdot x$, we define for every set X a function $\mathbb{R} \times (\mathbb{R} \times X \times X) \to \mathbb{R} \times FX$ by the assignment $(r, (s, x', x)) \mapsto (r \cdot s, r \cdot x')$.

It is not difficult to see that the three coproduct components of the ℓ-interpretation $b : \mathbb{R} \times C + C \times C + \mathbb{R} \times C \to C$ are indeed the desired three operations on streams: one just checks the commutativity of diagram (3.1).

According to Theorem 4.7 we can now define streams uniquely by giving a flat equation morphism $e : X \to HFX + C$.

Example 4.19. The system (4.3) of recursive equations gives rise to a flat equation morphism e where $X = \{x, y\}$, $e(x) = 0.y$ and $e(y) = 1.(y+x)$. This example does not make use of the parameters from C and thus it is already covered by Theorem 4.3. The solution of x is the stream $[0, 1, 1, 2, 3, 5, 8, \dots]$ of Fibonacci numbers.

We give another simple example using parameters and illustrating the compositionality of recursive equations:

Example 4.20. Consider the system

$$\begin{aligned} y_0 &= 1.(y_0 + y_1) \\ y_1 &= [2, 2, 2, 2, \dots] \end{aligned}$$

of recursive equations where the unique solution $y_0^\dagger$ of y_0 is given by the stream $[1, 3, 5, 7, 9, \dots]$ of odd natural numbers. By the modularity principle

from Remark 4.9 we can now use this stream in another system of recursive equations

$$x_0 = 0.(x_0 + x_1)$$
$$x_1 = y_0^\dagger$$

where the unique solution of x_0 is given by the stream $[0, 1, 4, 9, 16, \dots]$ of squares of natural numbers. This is easily seen by plugging in the stream $[1, 3, 5, 7, 9, \dots]$ for $y_0^\dagger$ and then solving the system of recursive equations. By the compositionality property of CIAs (see Remark 4.9) we could have also defined the stream $[0, 1, 4, 9, 16, \dots]$ as a solution of x_0 directly in one system of recursive equations of the form

$$x_0 = 0.(x_0 + x_1)$$
$$x_1 = 1.(y_0 + y_1)$$
$$y_0 = 1.(y_0 + y_1)$$
$$y_1 = [2, 2, 2, 2, \dots].$$

This is not as easy to see as above.

We remark that the advantage of compositionality grows with the number of recursive definitions that are composed. The reader is invited to turn the two-step definition of the stream $[0, 1, 4, 9, 16, \dots]$ of Example 4.20 into a three-step definition by defining first the stream $[2, 2, 2, 2, \dots]$ recursively as the unique solution of a variable z_0 and using z_0 as a parameter in the equation for y_1.

It should be mentioned that different formats for the recursive definition of streams were proposed by Rutten. First, in [Rut05a] *behavioral differential equations* are used; and second, in [Rut05b] streams are defined by *stream circuits* which are also called *(signal) flow graphs*. Since both formats extend to the recursive definition of stream functions, we shall discuss them in detail later in Section 5.2.1. The definition of streams is then the special case of the definition of nullary stream functions.

4.2.2 Infinite Trees

Silva and Rutten [SR10] have developed a calculus for infinite trees[1] similar to Rutten's stream calculus [Rut05a]. They provide *behavioral differential equations* for the recursive definition of infinite trees and prove a unique

[1] More precisely, Silva and Rutten work with complete binary trees whose nodes are labeled in $\mathbb{R}$. For the sake of brevity, we continue using the term "infinite trees".

solution theorem for them. Since this definition format extends to recursive definitions of tree functions, we shall discuss it in detail later in Section 5.2.2. Here we only give two examples of recursive definitions of trees in our setting.

Let $HX = X \times \mathbb{R} \times X$. The final coalgebra (C, c) for H is carried by the set C of all infinite binary trees with nodes labeled in $\mathbb{R}$, and its structure $c : C \to C \times \mathbb{R} \times C$ assigns to a tree t the triple (t_L, r, t_R) where r is the node label of the root of t and t_L and t_R are the trees rooted at the left-hand and right-hand child nodes of the root of t, see [MA86, SR10]. Consequently, the inverse $c^{-1} : C \times \mathbb{R} \times C \to C$ is the tree-tupling operation that constructs a tree by using the given real number as the root label and the given trees as subtrees rooted at the left-hand and right-hand children of the root. We denote this operation by the symbol $\int$.

For our tree definitions we shall need the operation $+$ of nodewise addition of trees (in the following it will always be clear from the context whether $+$ denotes addition of real numbers or trees). We define it by giving an abstract GSOS rule $\ell : K(H \times \mathrm{Id}) \to HF$, where $KX = X \times X$ is the functor associated to the signature of the binary operation $+$. For every set X, $\ell_X : (X \times \mathbb{R} \times X \times X)^2 \to FX \times \mathbb{R} \times FX$ is given by the assignment

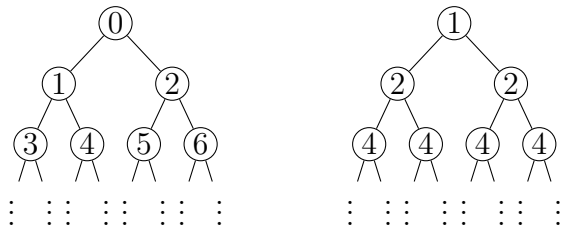
$$((x_L, r, x_R, x), (y_L, s, y_R, y)) \mapsto (x_L + y_L, r + s, x_R + y_R)\,.$$

It is easily verified that the ℓ-interpretation $b : C \times C \to C$ is indeed the desired operation $+$ on trees by checking the commutativity of diagram (3.1).

Example 4.21 (inspired by [SR10])**.** Consider the following system of recursive equations:

$$\begin{aligned} x_0 &= \textstyle\int(x_0 + y_0, 0, x_0 + y_0 + y_0) \\ y_0 &= \textstyle\int(y_0 + y_0, 1, y_0 + y_0) \end{aligned}$$

It corresponds to an ℓ-equation morphism $e : X \to HFX$ where $X = \{x_0, y_0\}$ and thus has a unique solution $e^\dagger$ by Theorem 4.3. This solution is given by the left-hand tree



labeled by the natural numbers for x_0 and by the right-hand tree labeled by the powers of 2 for y_0. The idea behind the definition of the left-hand tree

obviously is to use the right-hand tree. Thus it might be more convenient to write down a definition of the right-hand tree first and to use it in a definition of the left-hand one. This is possible by Remark 4.9 since our ℓ-equation morphism e also is a flat equation morphism $\mathsf{inl} \cdot e : X \to HFX + C$. We complete the latter by adding a new variable $x_1 = \int(y_0 + y_0, 1, y_0 + y_0)$ and decomposing it into two flat equation morphisms given by the systems

$$y_0 = \int(y_0 + y_0, 1, y_0 + y_0)$$

and

$$\begin{aligned} x_0 &= \int(x_0 + x_1, 0, x_0 + x_1 + x_1) \\ x_1 &= y_0^\dagger \end{aligned}$$

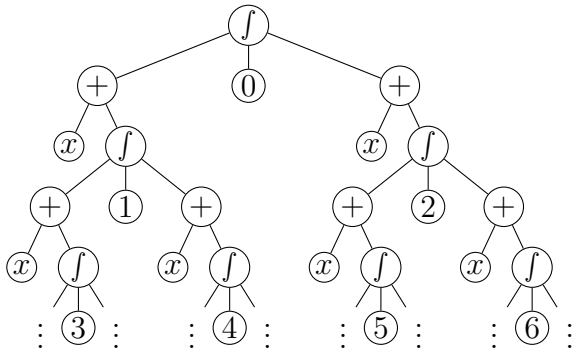
of recursive equations. These have unique solutions by Theorem 4.7 given by the above right-hand tree for y_0 and x_1, and by the above left-hand tree for x_0.

The next example illustrates Corollary 4.13:

Example 4.22. We consider the equation

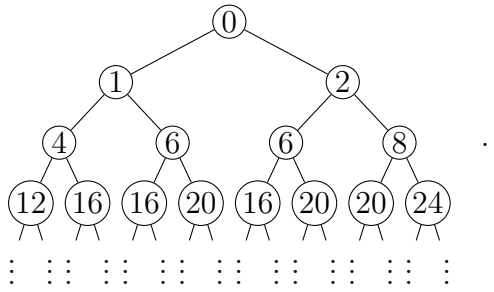
$$x = t$$

where $t \in T^{HF}(X + C)$ is the term given by the tree



.

Notice that this infinite term lies indeed in $T^{HF}(X+C)$: all infinite subterms become finite when their subterms starting with $\int$ are disregarded. Moreover, it lies in $HFT^{HF}(X+C)+C$ since the root node is labeled by the operation $\int$. Thus the above equation corresponds to a guarded equation morphism $e : X \to T^{HF}(X + C)$ where $X = \{x\}$, which has a unique solution by Corollary 4.13. This solution is given by the tree



4.2.3 CCS Processes

We shall be interested in Milner's calculus of communicating systems (or CCS, for short, see [Mil89]). Let κ be a regular cardinal and $\mathcal{P}_\kappa$ be the functor assigning to the set X the set of all subsets Y of X with $|Y| < \kappa$. Here we consider the functor $HX = \mathcal{P}_\kappa(A \times X)$ where A is some fixed alphabet of actions. Following Milner [Mil89], we assume that for every $a \in A$ we also have a complement $\bar{a} \in A$ (with $\bar{\bar{a}} = a$) and a special silent action $\tau \in A$.

We are now going to describe the final coalgebra for $\mathcal{P}_\kappa(A \times -)$. We start with a couple of definitions:

Definition 4.23. A *tree bisimulation* is a relation R on the node sets of two rooted, unordered (edge-labeled) trees such that

- the root nodes are related and
- for every pair $(n, o) \in R$ we have: for every child node n_i of n (via an edge labeled by a) there is a child node o_j of o (via an edge labeled by a) such that $(n_i, o_j) \in R$, and vice versa.

Two (edge-labeled) trees are called *bisimilar* if a tree bisimulation for them exists.

Definition 4.24. A rooted, unordered (edge-labeled) tree t is called *strongly extensional* if two subtrees rooted at distinct children of some node of t (where both edges to the children carry the same label) are never bisimilar.

Theorem 4.25 ([Wor05])**.** *The final coalgebra for the finite power set functor $\mathcal{P}_\mathrm{f}$ is carried by the set of all strongly extensional finitely branching trees.*

Theorem 4.26 ([Sch10a])**.** *The final coalgebra for the countable power set functor $\mathcal{P}_\mathrm{c}$ is carried by the set of all strongly extensional countably branching trees.*

The technique by which this result is obtained in loc. cit. generalizes to the functor $\mathcal{P}_\kappa(A \times -)$ from above:

Corollary 4.27. *Let κ be a regular cardinal. The final coalgebra (C, c) for the set functor $\mathcal{P}_\kappa(A \times -)$ is carried by the set C of all strongly extensional κ-branching trees with edges labeled in A.*

Furthermore, the structure map $c : C \to \mathcal{P}_\kappa(A \times C)$ of the final $\mathcal{P}_\kappa(A \times -)$-coalgebra assigns to a tree t the set of all pairs (a, t') where t' is a subtree of t rooted at a child of the root of t and where the edge between the roots of t and t' is labeled by a.

The elements of C can be considered as (denotations of) CCS agents (or CCS processes) modulo strong bisimilarity; for a concise definition of the latter see [Mil89]. Finally, it should be mentioned that CCS agents have been recognized as a final coalgebra much earlier by Aczel [Acz88]; however, in his work he assumes the antifoundation axiom.

Notice that the inverse $c^{-1} : \mathcal{P}_\kappa(A \times C) \to C$ assigns to a set $\{(a_i, E_i) \mid i < \kappa\}$ of pairs of actions and agents the agent $\sum_{i<\kappa} a_i.E_i$. In Milner's work [Mil89] the process combinators "$a.-$" (prefixing), "$|$" (parallel composition), "$\sum_{i<\kappa}$" (summation), "$-[f]$" (relabeling) and "$-\backslash L$" (restriction) are given by SOS rules. Let E, E', F, F' be agents and $a \in A$ some action, then these rules are:

$$\frac{E \xrightarrow{a} E'}{E|F \xrightarrow{a} E'|F} \qquad \frac{F \xrightarrow{a} F'}{E|F \xrightarrow{a} E|F'} \qquad \frac{E \xrightarrow{a} E' \quad F \xrightarrow{\bar{a}} F'}{E|F \xrightarrow{\tau} E'|F'} \quad (a \neq \tau)$$

$$\frac{}{a.E \xrightarrow{a} E} \qquad \frac{E_j \xrightarrow{a} E'_j}{(\sum_{i<\kappa} E_i) \xrightarrow{a} E'_j} \quad (j < \kappa) \tag{4.10}$$

$$\frac{E \xrightarrow{a} E'}{E[f] \xrightarrow{f(a)} E'[f]} \qquad \frac{E \xrightarrow{a} E'}{E\backslash L \xrightarrow{a} E'\backslash L} \quad (a, \bar{a} \notin L)$$

Now let K be the polynomial functor for the signature given by taking these combinators as operation symbols. It easily follows from the work in [Bar04] and [LPW04] that the above rules give an abstract GSOS rule $\ell : K(H \times \mathrm{Id}) \to HF$, and the ℓ-interpretation $b : KC \to C$ in C provides the desired operations on CCS agents (modulo strong bisimilarity). We explain how this abstract GSOS rule ℓ is obtained. The polynomial functor K is given on a set X by the coproduct of the following sets:

- $A \times X$ for agent expressions $a.x$, where $a \in A$,
- $\coprod_{n<\kappa} X^n$ for agent expressions $\sum_{i=1}^n x_i$, where $n < \kappa$,
- $X \times X$ for agent expressions $x_1|x_2$,

- $\coprod_f X$ for agent expressions $x[f]$, where f ranges over functions on the action set $A \setminus \{\tau\}$ with $\overline{f(a)} = f(\bar{a})$, and
- $\coprod_{L \subseteq A \setminus \{\tau\}} X$ for agent expressions $x \backslash L$.

The abstract GSOS rule $\ell : K(H \times \mathrm{Id}) \to HF$ is given by the SOS rules in (4.10) in terms of the components of the coproduct $K(H \times \mathrm{Id})$, i. e. for each combinator separately:

- $\ell_X(a, S, x) = \{(a, x)\}$ for prefixing $a.x$, where $S \subseteq A \times X$,
- $\ell_X((S_i, x_i)_{i<n}) = \bigcup_{i<n} S_i$ for summation $\sum_{i=1}^{n} x_i$ for every $n < \kappa$, where $(S_i)_{i<n}$ is a family of sets $S_i \subseteq A \times X$,
- $\ell_X(S_1, x_1, S_2, x_2)$ is given by the union of the three sets

$$\{(a, x|x_2) \mid (a, x) \in S_1\}, \quad \{(a, x_1|x) \mid (a, x) \in S_2\} \quad \text{and}$$

$$\{(\tau, x|y) \mid (a, x) \in S_1, (\bar{a}, y) \in S_2 \text{ for some } a \in A \setminus \{\tau\}\}$$

  for parallel composition $x_1|x_2$, where $S_1, S_2 \subseteq A \times X$,
- $\ell_X(S, x) = \{(f(a), y[f]) \mid (a, y) \in S\}$ for relabeling $x[f]$ (here we mean $f(\tau) = \tau$), where $S \subseteq A \times X$, and
- $\ell_X(S, x) = \{(a, y \backslash L) \mid (a, y) \in S, a, \bar{a} \notin L\}$ for restriction $x \backslash L$, where $S \subseteq A \times X$.

The form of these definitions is very similar to those given by Aczel [Acz88] in the setting of non-well-founded set theory. We already mentioned the ℓ-interpretation $b : KC \to C$ giving the desired operations on agents, and this gives the two new CIA structures for HF and FHF as in Theorems 4.7 and 4.8.

Remark 4.28. If we replaced the second component $\coprod_{n<\kappa} X^n$ of KX by $\mathcal{P}_\kappa X$ we still have an abstract GSOS rule. Furthermore, from diagram (3.1) it follows that in both cases the induced (binary) operation of summation (the corresponding component of $b : KC \to C$) is automatically commutative, associative and idempotent: these three laws that have to be proved in process theory come "for free" by encoding them in the abstract GSOS rule using the union operation.

Now let us recall Milner's solution theorem for CCS agents from [Mil89]. Suppose that E_i, $i \in I$, are agent expressions with the free variables x_i, $i \in I$. Consider the system of equations $x_i = E_i$ and suppose further that each variable x_j in each E_i, $i, j \in I$ is *weakly guarded*, i. e., it only occurs

within the scope of some prefix combinator $a.-$ in E_i. Then there is a unique solution of the system $x_i = E_i$. More precisely, recall strong bisimilarity of CCS agents from [Mil89] and let $E_i[\vec{P}/\vec{x}]$ denote simultaneous substitution of P_j for x_j for every j. Then we have

Theorem 4.29 ([Mil89])**.** *There exist, up to strong bisimilarity, unique CCS agents P_i such that P_i is strongly bisimilar to $E_i[\vec{P}/\vec{x}]$ for each $i \in I$.*

It is easy to see that this theorem is a consequence of our Theorem 4.5: a system $x_i = E_i$ where each variable is weakly guarded is essentially the same as a map $X \to FHFX$, where $X = \{x_i \mid i \in I\}$. Now our Theorem 4.8 generalizes Theorem 4.5 to flat equation morphisms $X \to FHFX+C$. These again have unique solutions in C. The extra summand C allows us to use constant agents in recursive specifications. So, for example, we can obtain the agent P as the unique solution of

$$x = a.(x|c) + b$$

and then use it in a system like

$$\begin{aligned} x &= b.(x + y) \\ y &= P \end{aligned}$$

which has a unique solution by Theorem 4.8.

4.2.4 Formal Languages

Consider the set functor $HX = X^A \times 2$, where $2 = \{0, 1\}$. Coalgebras for H are precisely the (possibly infinite) deterministic automata with input alphabet A. The final H-coalgebra $c : C \to HC$ consists of all formal languages with $c(L) = (\lambda a.L^a, i)$ where $i = 1$ iff the empty word ε is in L and where $L^a = \{w \mid aw \in L\}$, see [MA86].

To specify e. g. the intersection of formal languages by an abstract GSOS rule, let $KX = X \times X$ and let $\ell : K(H \times \mathrm{Id}) \to HF$ be induced by $\ell' : KH \to HK$ according to Remark 3.18 as follows: $\ell' : KH \to HK$ is given for every set X by $\ell'_X((f, i), (g, j)) = (\langle f, g\rangle, i \wedge j)$ where $\wedge$ denotes the "and"-operation on $\{0, 1\}$. Then the ℓ-interpretation is easily verified to be the intersection of formal languages.

Next we show how context-free grammars in Greibach normal form and their generated languages are special instances of flat equation morphisms $e : X \to FHFX + C$ and their unique solutions in C for a suitable functor K.

Definition 4.30 (see e. g. [HMU07])**.** A *context-free grammar* is a four-tuple $G = (A, N, P, S)$ where A is a nonempty finite set of *terminal* symbols, N a finite set of *non-terminal* symbols, $P \subseteq N \times (A + N)^*$ is a finite relation with elements called *production rules* of G, and $S \in N$ is the *starting symbol*. The language *generated* by a context-free grammar G is the set of all words over A that arise by starting with the string S and repeatedly substituting substrings according to the production rules of the grammar, and eliminating ε from the string whenever it occurs.

As usual we write $n \to w$ for $(n, w) \in P$.

Definition 4.31 ([Gre65])**.** A context-free grammar G is in *Greibach normal form* (*GNF*, for short) if all its production rules are of the form $n \to aw$ where $a \in A$ and $w \in N^*$.

To see that context-free grammars in GNF yield flat equation morphisms we consider the empty language $\emptyset$, the empty-word language $\{\varepsilon\}$ and union $\cup$ and concatenation $\cdot$ of languages as given operations. More precisely, let $KX = 1 + 1 + X \times X + X \times X$ be the polynomial functor corresponding to the signature of the operations $\emptyset$, ε, $\cup$ and $\cdot$, and let $\ell : K(H \times \mathrm{Id}) \to HF$ be the abstract GSOS rule given by the following assignments:

$$\begin{aligned} \emptyset &\mapsto ((\emptyset)_{a\in A}, 0) \\ \varepsilon &\mapsto ((\emptyset)_{a\in A}, 1) \\ ((x_a), j, x) \cup ((y_a), k, y) &\mapsto ((x_a \cup y_a), j \vee k) \\ ((x_a), j, x) \cdot ((y_a), k, y) &\mapsto ((t_a), j \wedge k) \end{aligned} \tag{4.11}$$

where

$$t_a = \begin{cases} (x_a \cdot y) \cup y_a & \text{if } j = 1 \\ x_a \cdot y & \text{else}\,. \end{cases}$$

Commutativity of diagram (3.1) is easily checked, thus the ℓ-interpretation $b : KC \to C$ is given as desired. By Theorem 4.8 we obtain the CIA structure $k' : FHFC \to C$. Now observe that a flat equation morphism assigns to each $x \in X$ either an element $e(x) \in C$ or $e(x)$ corresponds to a term of given operations on HFX.

We now show how to construct a flat equation morphism for any context-free grammar in GNF.

Construction 4.32. Let $G = (A, N, P, S)$ be a context-free grammar in GNF. We define a flat equation morphism $e_G : N \to FHFN + C$ as follows: for $n \in N$ for which there is no production rule $n \to aw$ in P we take

$e_G(n) = \emptyset$, the constant term in $F(HFN)$. Otherwise we define for each production rule $n \to aw$ with n on the left-hand side the term $t_r \in HFN$ as

$$t_r = \begin{cases} a.(n_1 \cdot n_2 \cdot \cdots \cdot n_k) & w = n_1 n_2 \cdots n_k, k \geq 1 \\ a.\varepsilon & w = \varepsilon \end{cases}$$

using the concatenation operation. Here the notation $a.t$ where $t \in FN$ is used as a shortcut for $((t_b), 0) \in HFN$ where (t_b) is the A-tuple with $t_a = t$ and $t_b = \emptyset$ for every $b \in A \backslash \{a\}$; thus we see that $t_r \in HFN$. We define $e_G(n)$ as (the term in $FHFN$ representing) the "union" of all terms corresponding to the right-hand sides of the production rules $n \to r_i$, $i = 1, \ldots, l$, in P:

$$e_G(n) = t_{r_1} \cup t_{r_2} \cup \cdots \cup t_{r_l} .$$

Notice that e_G does not make use of the parameters in C.

It is not difficult to see that the language generated by the grammar G is precisely the language $e_G^\dagger(S)$, where S is the starting symbol of G. So as a consequence of Theorem 4.8 we see that the language generated by G arises as the unique solution of the flat equation morphism e_G. Thus we obtained a denotational semantics of grammars in GNF which coincides with the operational semantics given by language generation.

Similarly, it is possible to translate right-linear grammars (which are a special case of context-free grammars generating regular languages) into flat equation morphisms using the empty and empty-word languages as well as union of languages as the given operations. Again Theorem 4.8 implies that there is a unique solution which yields the language generated by the given right-linear grammar as one can see from the translation.

4.2.5 Non-well-founded Sets

For background on non-well-founded sets and the antifoundation axiom (AFA), the reader can consult the books [Acz88, BM96]. We work here on the category Class of classes and functions. Assuming the axiom of choice (AC) for classes, the results of Section 4.1.2 hold true for Class since every endofunctor of Class has final coalgebras and free algebras, see [AMV04].

Consider $\mathcal{P} : \mathsf{Class} \to \mathsf{Class}$ taking a class X to the class $\mathcal{P}X$ of sub*sets* of X. AFA is equivalent to the assertion that (V, c) is a final $\mathcal{P}$-coalgebra, where V is the class of all non-well-founded sets, and $c : V \to \mathcal{P}V$ takes a set and considers it a set of sets, see [Acz88]. That is, c is the identity function.

The following examples of this subsection were originally given by Moss. Let us note some natural transformations:

$$\begin{array}{lll} p : \mathcal{P} \to \mathcal{P}\mathcal{P} & op : \mathrm{Id} \times \mathrm{Id} \to \mathcal{P}\mathcal{P} & cp : \mathcal{P} \times \mathcal{P} \to \mathcal{P}(\mathrm{Id} \times \mathrm{Id}) \\ p_X(x) = \mathcal{P}(x) & op_X(x,y) = \{\{x\},\{x,y\}\} & cp_X(x,y) = x \times y \end{array}$$

Also note that c^{-1} is the operation on V taking a family $v \subseteq V$ of sets to the set $\{\, w \mid w \in v \,\}$.

We will now define three additional operations on V: the powerset operation $b_1 : v \mapsto \{\, w \mid w \subseteq v \,\}$, the Kuratowski pair $b_2 : (v,w) \mapsto \{\{\, v \,\}, \{\, v, w\}\}$ and the cartesian product $b_3 : (v,w) \mapsto v \times w$. So let K be the functor $\mathrm{Id} + (\mathrm{Id} \times \mathrm{Id}) + (\mathrm{Id} \times \mathrm{Id}) + \mathcal{P} + \mathcal{P}^2$; its first three components represent (the type of) our three desired operations, the fourth component, $\mathcal{P}$, represents c^{-1} and the fifth one represents $c^{-1} \cdot \mathcal{P}c^{-1}$. The latter two are needed for the definition of the former three below. We write the coproduct injections of K as $\mathsf{inj}_1, \ldots, \mathsf{inj}_5$. We define a natural transformation $\ell' : K\mathcal{P} \to \mathcal{P}K$ componentwise, using

$$\begin{array}{rl} \mathcal{P} & \xrightarrow{\;p\;} \mathcal{P}\mathcal{P} \xrightarrow{\;\mathcal{P}\mathsf{inj}_4\;} \mathcal{P}K \\ \mathcal{P} \times \mathcal{P} & \xrightarrow{\;op\mathcal{P}\;} \mathcal{P}\mathcal{P}\mathcal{P} \xrightarrow{\;\mathcal{P}\mathsf{inj}_5\;} \mathcal{P}K \\ \mathcal{P} \times \mathcal{P} & \xrightarrow{\;cp\;} \mathcal{P}(\mathrm{Id} \times \mathrm{Id}) \xrightarrow{\;\mathcal{P}\mathsf{inj}_2\;} \mathcal{P}K \end{array} \qquad \begin{array}{l} \mathcal{P}\mathcal{P} \xrightarrow{\;\mathcal{P}\mathsf{inj}_4\;} \mathcal{P}K \\ \mathcal{P}\mathcal{P}\mathcal{P} \xrightarrow{\;\mathcal{P}\mathsf{inj}_5\;} \mathcal{P}K \end{array}$$

Then ℓ' yields an abstract GSOS rule

$$\ell = (\, K(\mathcal{P} \times \mathrm{Id}) \xrightarrow{K\pi_0} K\mathcal{P} \xrightarrow{\ell'} \mathcal{P}K \xrightarrow{\mathcal{P}\kappa} \mathcal{P}F \,)\,.$$

Let $b : KV \to V$ be the ℓ-interpretation in V. Let us write $b_1, \ldots, b_5$ for the components of $b : KV \to V$, i. e. $b_i = b \cdot (\mathsf{inj}_i)_V$, $1 \le i \le 5$. To obtain explicit formulas for these, we use diagram (3.1) and the above definitions:

$$\begin{array}{ll} c \cdot b_1 = \mathcal{P}b_4 \cdot p_V \cdot c & c \cdot b_4 = \mathcal{P}b_4 \cdot \mathcal{P}c \\ c \cdot b_2 = \mathcal{P}b_5 \cdot op_{\mathcal{P}V} \cdot (c \times c) & c \cdot b_5 = \mathcal{P}b_5 \cdot \mathcal{P}^2 c \\ c \cdot b_3 = \mathcal{P}b_2 \cdot cp_V \cdot (c \times c) & \end{array}$$

We check easily that $b_4 = c^{-1}$ and $b_5 = c^{-1} \cdot \mathcal{P}c^{-1}$ satisfy the last two equations. From these we see that

$$b_1 = c^{-1} \cdot \mathcal{P}c^{-1} \cdot p_V \cdot c\,, \qquad b_2 = c^{-1} \cdot \mathcal{P}(c^{-1} \cdot \mathcal{P}c^{-1}) \cdot op_{\mathcal{P}V} \cdot (c \times c)\,,$$

and

$$b_3 = c^{-1} \cdot \mathcal{P}b_2 \cdot cp_V \cdot (c \times c)\,.$$

In words, b_4 and b_5 are the identity functions, and b_1, b_2 and b_3 are as desired.

By Theorem 4.7, we have a CIA structure $(V, c^{-1} \cdot \mathcal{P}\widehat{b})$ for the composite $\mathcal{P}F$. Now we may solve systems of recursive equations which go beyond what one finds in the standard literature on non-well-founded sets [Acz88, BM96]:

Example 4.33. One may solve the system

$$\begin{aligned} x &= \{\mathcal{P}(y)\} \\ y &= \{y \times y, z\} \\ z &= \emptyset\,, \end{aligned}$$

uniquely, which gives rise to a flat equation morphism $X \to \mathcal{P}FX + V$ where $X = \{x, y, z\}$.

For our next example, we now define the unordered pair operation $un : (v, w) \mapsto \{v, w\}$ by giving another abstract GSOS rule $\ell : K(\mathcal{P} \times \mathrm{Id}) \to \mathcal{P}F$, where $KX = X \times X + \mathcal{P}X$. Let the natural transformation $u : \mathrm{Id} \times \mathrm{Id} \to \mathcal{P}$ be given by $u_X(x, y) = \{\, x, y \,\}$. We define a natural transformation $\ell' : K\mathcal{P} \to \mathcal{P}K$ by

$$\ell' = (\, K\mathcal{P} = \mathcal{P} \times \mathcal{P} + \mathcal{P}\mathcal{P} \xrightarrow{[u\mathcal{P}, \mathrm{id}]} \mathcal{P}\mathcal{P} \xrightarrow{\mathcal{P}\mathsf{inr}} \mathcal{P}(\mathrm{Id} \times \mathrm{Id} + \mathcal{P}) = \mathcal{P}K\,)\,.$$

and let ℓ be the induced abstract GSOS rule according to Remark 3.18. We get $b : KV \to V$ making the diagram (3.1) commute which can be simplified to the following commutative diagram (by using $\widehat{b} \cdot \kappa_V = b$, see below Notation 3.4):

$$\begin{array}{ccccc} V^2 + \mathcal{P}V & \xrightarrow{c^2 + \mathcal{P}c} & (\mathcal{P}V)^2 + \mathcal{P}\mathcal{P}V & \xrightarrow{\ell'_V} & \mathcal{P}(V^2 + \mathcal{P}V) \\ {\scriptstyle b}\downarrow & & & & \downarrow{\scriptstyle \mathcal{P}b} \\ V & & \xrightarrow[c]{} & & \mathcal{P}V \end{array} \tag{4.12}$$

It is not difficult to verify that the right-hand component of $b : V^2 + \mathcal{P}V \to V$ is $c^{-1} : \mathcal{P}V \to V$. It follows that the left-hand component of $b : V^2 + \mathcal{P}V \to V$ is the unordered pair operation $un : V^2 \to V$. By Theorem 4.7, (V, k) is a CIA for $\mathcal{P}F$, where $k : \mathcal{P}FV \to V$ is $c^{-1} \cdot \mathcal{P}\widehat{b}$.

Example 4.34. As a concrete example of a flat equation morphism, let $X = \{x, y\}$, and let $e : X \to \mathcal{P}FX + V$ be given by the system

$$\begin{aligned} x &= \{un(y, y)\} \\ y &= \{\{x\}\} \end{aligned}$$

of equations. Its unique solution $e^\dagger$ consists of sets v and w satisfying $v = \{\{w\}\}$ and $w = \{\{v\}\}$. The only non-well-founded sets with this property are the infinitely nested singleton sets $v = w = \{\{\{\dots\}\}\}$.

4.3 Excursion: Finite Systems of Recursive Equation

In Section 4.1, we have proved algebras $k : HFC \to C$ and $k' : FHFC \to C$ on a final coalgebra C to be CIAs. Thus, arbitrary flat equation morphisms and guarded equation morphisms can be solved uniquely. In the category Set this means that we can solve recursive specifications with finitely and infinitely many variables. At the end of Section 4.1.1 we argued that using the simple subformat $e : X \to HX$ of a flat equation morphism, we can already define every element of the final H-coalgebra C.

In practice, one only encounters specifications with finitely many variables. It is an interesting question which elements of the final H-coalgebra C are definable by such specifications; it turns out that this depends on the ℓ-interpretation $b : KC \to C$, i. e. on the operations available.

In this section, we first recall the concept of an iterative algebra from [AMV06b] which abstractly (not only for Set) captures algebras in which "finite" specifications have unique solutions. Then we describe important subsets of final coalgebras definable by certain choices of algebraic operations for three of the five examples from Section 4.2: streams, CCS processes and formal languages.

Recall (e. g. from [AR94]) the notion of a *filtered colimit*. Also recall that a category ist called *cocomplete* if all small colimits exist.

Definition 4.35 ([AR94])**.** Let $\mathcal{C}$ be a category. A *finitely presentable object* of $\mathcal{C}$ is an object X for which the hom-functor $\mathcal{C}(X, -) : \mathcal{C} \to \mathsf{Set}$ is finitary, i. e. it preserves all filtered colimits. $\mathcal{C}$ is called a *locally finitely presentable category* if it is cocomplete and there exists a set of finitely presentable objects whose closure under filtered colimits is $\mathcal{C}$.

Examples 4.36.

1. Set is locally finitely presentable. The finitely presentable objects are precisely the finite sets.

2. The category Vec of all real vector spaces and linear transformations between them is locally finitely presentable. The finitely presentable objects are precisely the finite dimensional vector spaces.

Definition 4.37 ([AMV06b])**.** Let H be a finitary endofunctor on a locally finitely presentable category. A *finitary flat equation morphism* (with parameters in A) is a morphism $e : X \to HX + A$ where X is finitely presentable. A *solution* of e in an H-algebra (A, a) is a morphism $e^\dagger : X \to A$ making diagram (2.1) commute. An *iterative algebra for H* is an H-algebra (A, a) in which every finitary flat equation morphism with parameters in A has a unique solution.

Examples 4.38. 1. Every H-CIA (H as in Definition 4.37) is an iterative algebra.

2. The algebra $((0,\infty],+)$ for the functor $HX = X \times X$ is iterative, but not a CIA, see [AMV06b].

Remarks 4.39. 1. In our examples below we work with finitary endofunctors H on the categories Set and Vec; these two categories have the property that strong quotients of finitely presentable objects are again finitely presentable. Under this assumption, it was proved in Proposition 4.6 of [AMV03] in conjunction with Theorem 3.3 of [AMV06b] that the carrier of the initial iterative H-algebra can be formed as a certain colimit which is in case of Set and Vec the union

$$\bigcup_{\substack{e:X\to HX \\ X \text{ finitely presentable}}} e^\dagger[X]\,,$$

where $e^\dagger : X \to C$ is the unique homomorphism from $e : X \to HX$ into the final H-coalgebra $c : C \to HC$.

2. For polynomial set functors H_Σ (for a finitary signature Σ), the initial iterative H_Σ-algebra is given by the set of all rational Σ-trees together with tree tupling, see Example 3.6 in [AMV06b]. Here a Σ-tree is called *rational* if it has (up to isomorphism) finitely many subtrees.

4.3.1 Eventually Periodic and Rational Streams

Recall from Section 4.2.1 that streams form the final coalgebra $(C, c) = (\mathbb{R}^\omega, \langle \mathsf{hd}, \mathsf{tl}\rangle)$ for the set functor $HX = \mathbb{R} \times X$. Given a stream σ, we call the streams $\mathsf{tl}^n(\sigma)$ for every $n \in \mathbb{N}$ its *derivatives*.

Definition 4.40. A stream is called *eventually periodic* (or *ultimately periodic*) if it has only finitely many distinct derivatives. We denote the set of all eventually periodic streams by $\mathbb{R}^\omega_{\mathrm{ep}}$.

Theorem 4.41. *The union of all images of the solutions of finitary flat equation morphisms with parameters from $\mathbb{R}^\omega_{\mathrm{ep}}$ in the initial CIA (C, c^{-1}) of streams form precisely the set of all eventually periodic streams, i. e.*

$$\bigcup_{\substack{e:X\to HX+\mathbb{R}^\omega_{\mathrm{ep}} \\ X \text{ finitely presentable}}} e^\dagger[X] = \mathbb{R}^\omega_{\mathrm{ep}}\,.$$

Proof. This follows from Remarks 4.39(1) and (2):

First observe that streams can be viewed as Σ-trees for the stream functor $HX = \mathbb{R} \times X$, which are infinite chains where every node is labeled in $\mathbb{R}$, and the rational such trees obviously correspond precisely to the eventually periodic streams. Thus by Remark 4.39(2) $\mathbb{R}^\omega_{\mathrm{ep}}$ is the carrier of the initial iterative H-algebra.

Second, by the formula from Remark 4.39(1) for the carrier of the initial iterative H-algebra we see that the images of solutions of finitary flat equation morphisms without parameters (i. e. H-coalgebras $e : X \to HX$) form precisely the set $\mathbb{R}^\omega_{\mathrm{ep}}$.

And third, adding parameters in $\mathbb{R}^\omega_{\mathrm{ep}}$ (i. e. working with finitary flat equation morphisms $e : X \to HX + \mathbb{R}^\omega_{\mathrm{ep}}$) does not change the situation: for each of the finitely many $r_i \in \mathbb{R}^\omega_{\mathrm{ep}}$, $1 \leq i \leq n$, which occur on the right-hand side of the system e of equations, we can find an H-coalgebra $e_i : Y_i \to HY_i$ with finitely presentable Y_i such that there is $y_i \in Y_i$ with $e^\dagger(y_i) = r_i$. Indeed, this follows from the formula from Remark 4.39(1). We can form an H-coalgebra $\bar{e} : X + Y_1 + \cdots + Y_n \to H(X + Y_1 + \cdots + Y_n)$ by composing the equation systems e (with r_i replaced by the right-hand sides of the equations for the variables y_i for every $1 \leq i \leq n$) and e_i, $1 \leq i \leq n$. Since $X + Y_1 + \cdots + Y_n$ is finitely presentable, $\bar{e}$ is solved in $\mathbb{R}^\omega_{\mathrm{ep}}$. In particular, $\bar{e}^\dagger[X] \subseteq \mathbb{R}^\omega_{\mathrm{ep}}$, and since by the construction of $\bar{e}$ we have $e^\dagger = \bar{e}^\dagger \cdot \mathsf{inl}$, we conclude $e^\dagger[X] \subseteq \mathbb{R}^\omega_{\mathrm{ep}}$. On the other hand side, every finitary flat equation morphism without parameters can be viewed as a finitary flat equation morphism with parameters using the left injection which has the same solution, see Remark 4.10. This means that using parameters we still have all regular languages in the union of images of solutions. □

However, as we shall prove next, adding the operations of componentwise stream addition $+$ and scalar multiplication $r \cdot -$ (see Section 4.2.1), we obtain a strictly larger subset of streams as images of solutions of equations $X \to HFX$ where X is finite and F is the free monad over the polynomial functor $KX = X \times X + \mathbb{R} \times X$ corresponding to the two operations on streams.

Assumption 4.42. Throughout the rest of Section 4.3.1, we let $KX = X \times X + \mathbb{R} \times X$ and denote by F the free monad over K. From Section 4.2.1 we already know how to define an abstract GSOS rule $\ell : K(H \times \mathrm{Id}) \to HF$ (leaving out the coproduct component of K for the prefixing operation) whose interpretation yields the operations of componentwise stream addition and scalar multiplication for streams. We denote by (C, k) the CIA we obtain from Theorem 4.7 applied to ℓ.

Recall (e. g. from [Rut05b]) that there is a binary operation $\times$ on streams, called the *convolution product*, and a unary operation $(-)^{-1}$ giving the *inverse* of a stream σ w. r. t. $\times$ whenever $\mathsf{hd}(\sigma) \neq 0$. Since we shall not use their precise definitions in the present subsection, we refer the reader to Example 5.35 below for the definition of $\times$ and to [Rut05b] for the definition of $(-)^{-1}$.

Definition 4.43 ([Rut05b])**.** A stream is called *polynomial* if only finitely many components are different from 0. A stream ρ is called *rational* if it has the form $\rho = \sigma \times \tau^{-1}$ for polynomial streams σ and τ where $\mathsf{hd}(\tau) \neq 0$. We denote the set of all rational streams by $\mathbb{R}^{\omega}_{\mathrm{ra}}$.

Examples 4.44.

1. Every eventually periodic stream is rational, see Remark 2.4 of [Rut08].

2. The stream $[1, 2, 4, 8, 16, 32, \dots]$ of powers of 2 is rational since it can be expressed as $[1, 0, 0, 0, \dots] \times [1, -2, 0, 0, \dots]^{-1}$, see Example 2.6 of loc. cit.; but clearly it is not eventually periodic.

3. The stream $[1, 1, 0, 1, 0, 0, 1, 0, 0, 0, \dots]$ is not rational, see Example 5.4 of loc. cit.

Now observe that $\mathbb{R} \times -$ is not only a functor on Set, but also a functor on Vec: $\mathbb{R}$ is a one-dimensional vector space, and the product with any n-dimensional vector space V is the $(n+1)$-dimensional vector space $\mathbb{R} \times V$.

The final coalgebra for $\mathbb{R} \times - : \mathsf{Vec} \to \mathsf{Vec}$ is $(\mathbb{R}^{\omega}, \langle \mathsf{hd}, \mathsf{tl} \rangle)$, see Section 4 in [Rut08]. It is the same one as for $\mathbb{R} \times - : \mathsf{Set} \to \mathsf{Set}$ except that $\mathbb{R}^{\omega}$ is completed to the vector space $(\mathbb{R}^{\omega}, +, \cdot)$ where $+$ is the componentwise stream addition and $\cdot$ is the scalar multiplication for streams. As a special case of the results in loc. cit. we have the following

Theorem 4.45 (see [Rut08])**.** *For every coalgebra (V, v) for the functor $\mathbb{R} \times -$ on Vec with finite dimensional V and for the final linear transformation $f : V \to \mathbb{R}^{\omega}$ it holds $f[V] \subseteq \mathbb{R}^{\omega}_{\mathrm{ra}}$. Moreover, every rational stream lies in the image $f[V]$ for the final linear transformation $f : V \to \mathbb{R}^{\omega}$ for some $(\mathbb{R} \times -)$-coalgebra (V, v) with finite dimensional V.*

Next we provide a translation of every finitary flat equation morphism $e : X \to HFX + \mathbb{R}^{\omega}_{\mathrm{ra}}$ to a coalgebra (W, w) for $\mathbb{R} \times - : \mathsf{Vec} \to \mathsf{Vec}$ where W is finite dimensional.

Construction 4.46. Let $e : X \to HFX + \mathbb{R}^{\omega}_{\mathrm{ra}}$ be a finitary flat equation morphism and let $b_X : FX \to X$ be the extension of the vector space operations of X to terms from FX. First consider the largest subset $X_0 \subseteq X$ with

$e[X_0] \subseteq HFX$. These are the variables x with $e(x) = (r, t)$ where $t \in FX$ is a term over X built from the operations $+$ and $\cdot$. Second we consider all other variables $x_i \in X_1$, $i < |X_1|$, where $e[X_1] \subseteq \mathbb{R}^\omega_{\mathrm{ra}}$. For these variables $e(x_i) = \sigma_i$ is a rational stream. It follows from Theorem 5.4 of [Rut08] that derivatives of rational streams are rational again, so we have $\sigma_i = r_i.\sigma_i'$ for some $\sigma_i' \in \mathbb{R}^\omega_{\mathrm{ra}}$. According to the second part of Theorem 4.45 we have a corresponding $(\mathbb{R} \times -)$-coalgebra (V_i, v_i) with finite dimensional V_i and a vector $w_i \in V_i$ such that $f_i(w_i) = \sigma_i'$ for the final linear transformation $f_i : V_i \to \mathbb{R}^\omega$. We denote by $b_i : FV_i \to V_i$ the extension of the vector space operations to terms from FV_i.
We construct the $(\mathbb{R} \times -)$-coalgebra (W, w) as follows: the vector space W has the base $X + \coprod_{i<|X_1|} B_i$ where B_i is a base of V_i, i. e. $\dim W = |X| + \sum_{i<|X_1|} \dim V_i$. Then clearly W is finite dimensional since X is and all of the V_i are. We denote by $\mathsf{inj} : X \to W$ the injection mapping variables to the corresponding unit vectors; similarly we have injections $\mathsf{inj}_i : B_i \to W$, $i < |X_1|$, for the other base vectors of W. We define the extension $b_W : FW \to W$ of the vector space operations of W to terms from FW by $(b_W \cdot F\mathsf{inj})(s) = (\mathsf{inj} \cdot b_X)(s)$ for $s \in FX$ and by $(b_W \cdot F\mathsf{inj}_i)(r) = (\mathsf{inj}_i \cdot b_i)(r)$ for $r \in FV_i$, $i < |X_1|$. Now b_W is completely defined since we defined it on the bases of all of the subspaces forming W. For every $x \in X_0$ we define $(w \cdot \mathsf{inj})(x) = (r, v_t)$ where $v_t = (b_W \cdot F\mathsf{inj})(t)$ is the vector corresponding to the result of the term t when the variables from X are viewed as unit vectors. For every $x_i \in X_1$ we define $(w \cdot \mathsf{inj})(x_i) = (r_i, w_i)$; and for the unit vectors u of the spaces V_i we define $w(u) = v_i(u)$. Now the $(\mathbb{R} \times -)$-coalgebra (W, w) is completely defined since we defined w on a base of W and w is required to be a linear transformation.

Lemma 4.47. *Let (W, w) be the translation of e according to Construction 4.46. Then the final map $f : W \to \mathbb{R}^\omega$ precomposed with inj from Construction 4.46 is the solution of e in the CIA (C, k).*

Proof. We consider the finality diagram

$$\begin{array}{ccc} W & \xrightarrow{f} & \mathbb{R}^\omega \\ {\scriptstyle w}\downarrow & & \downarrow{\scriptstyle c} \\ \mathbb{R} \times W & \xrightarrow[\mathrm{id} \times f]{} & \mathbb{R} \times \mathbb{R}^\omega \end{array},$$

invert c and precompose it with inj. For the variables $x_i \in X_1$ from Con-

struction 4.46 we obtain

$$\begin{aligned}(f \cdot \mathsf{inj})(x_i) &= (c^{-1} \cdot (\mathrm{id} \times f) \cdot w \cdot \mathsf{inj})(x_i) \\ &= (c^{-1} \cdot (\mathrm{id} \times f))(r_i, w_i) \\ &= c^{-1}(r_i, \sigma_i') \\ &= \sigma_i \\ &= e(x_i) \\ &= ([c^{-1}, \mathrm{id}] \cdot (Hb + \mathrm{id}) \cdot (HF(f \cdot \mathsf{inj}) + \mathrm{id}) \cdot e)(x_i)\,.\end{aligned}$$

where all equalities follow from Construction 4.46. Here $b : F\mathbb{R}^\omega \to \mathbb{R}^\omega$ is the extension of the vector space operations of $\mathbb{R}^\omega$ to terms from $F\mathbb{R}^\omega$. For the variables $x \in X_0$ we have

$$\begin{aligned}(f \cdot \mathsf{inj})(x) &= (c^{-1} \cdot (\mathrm{id} \times f) \cdot w \cdot \mathsf{inj})(x) \\ &= (c^{-1} \cdot (\mathrm{id} \times f))(r, v_t) \\ &= c^{-1}(r, f(v_t)) \\ &= c^{-1}(r, (f \cdot b_W \cdot F\mathsf{inj})(t)) \\ &= c^{-1}(r, (b \cdot F(f \cdot \mathsf{inj}))(t)) \\ &= (c^{-1} \cdot Hb \cdot HF(f \cdot \mathsf{inj}))(r, t) \\ &= ([c^{-1}, \mathrm{id}] \cdot (Hb + \mathrm{id}) \cdot (HF(f \cdot \mathsf{inj}) + \mathrm{id}) \cdot e)(x)\end{aligned}$$

using Construction 4.46 and linearity $f \cdot b_W = b \cdot Ff$ of f. Thus uniqueness of solutions of e in (C, k) yields $e^\dagger = f \cdot \mathsf{inj}$. □

Theorem 4.48. *The union of all images of the solutions of finitary flat equation morphisms with parameters in $\mathbb{R}^\omega_{\mathrm{ra}}$ in the CIA (C, k) (using the operations $+$ and $r \cdot -$ defined by ℓ) form precisely the set of all rational streams, i. e.*

$$\bigcup_{\substack{e : X \to HFX + \mathbb{R}^\omega_{\mathrm{ra}} \\ X \text{ finitely presentable}}} e^\dagger[X] = \mathbb{R}^\omega_{\mathrm{ra}}\,.$$

Proof. We prove the desired equality of sets by considering the two subset inclusions separately:

$\subseteq$ For every finitary flat equation morphism $e : X \to HFX + \mathbb{R}^\omega_{\mathrm{ra}}$ we can perform Construction 4.46 and obtain the $(\mathbb{R} \times -)$-coalgebra (W, w). Since it is immediate from this construction that W is finite dimensional, it follows from Theorem 4.45 that $f[W] \subseteq \mathbb{R}^\omega_{\mathrm{ra}}$ for the final linear transformation $f : W \to \mathbb{R}^\omega$. Thus $e^\dagger[X] \subseteq \mathbb{R}^\omega_{\mathrm{ra}}$ by Lemma 4.47, i. e. all solutions of variables from X are rational streams.

$\supseteq$ This direction is trivial since we obtain every rational stream $\sigma \in \mathbb{R}^\omega_{\mathrm{ra}}$ as a solution of the finitary flat equation morphism $e : X \to HFX + \mathbb{R}^\omega_{\mathrm{ra}}$ given by $X = \{x\}$ and $e(x) = \sigma$. □

Remark 4.49. Theorem 4.48 also follows from Remark 4.39(1) and [Mil10]: in loc. cit. it was shown (Corollary III.15 and Example III.16(3)) that the initial iterative algebra for the functor $\mathbb{R} \times -$ on Vec is given by the subcoalgebra of $(\mathbb{R}^\omega, \langle \mathsf{hd}, \mathsf{tl} \rangle)$ given by all rational streams. This can be used (instead of Remark 4.39(2)) in a proof similar to the one of Theorem 4.41.

4.3.2 Rational (or Regular) CCS Processes

Whereas in Section 4.2.3 we worked with the functor $HX = \mathcal{P}_\kappa(A \times X)$ for an arbitrary regular cardinal κ and an arbitrary set A, in this section we restrict ourselves to $\kappa = \omega$ and finite sets A and work with the functor $HX = \mathcal{P}_{\mathrm{f}}(A \times X)$ where $\mathcal{P}_{\mathrm{f}}$ is the finite powerset functor. This makes sense since we want to investigate flat equation morphisms with only a finite set X of variables. As a consequence one can restrict to a finite alphabet A by quotienting the original infinite alphabet so that actions that always occur in parallel and have the same subexpression (i. e. as a subterm $a.x + b.x$) are modeled by a single action without loss of any information. Now, since the variable set X and the alphabet A are finite, $A \times X$ and all its subsets are finite, so it suffices to consider the finite powerset functor.

Recall from Section 4.2.3 that the final coalgebra C for $\mathcal{P}_{\mathrm{f}}(A \times -)$ is carried by the set of all strongly extensional finitely branching (unordered) trees with edges labeled in A, and these trees can be considered as the CCS agents or CCS processes modulo strong bisimilarity. Observe that the functor $\mathcal{P}_{\mathrm{f}}(A \times -)$ is finitary; the initial iterative algebra for $\mathcal{P}_{\mathrm{f}}(A \times -)$ is carried by the set C_{ra} of all rational trees, i. e. trees which have only finitely many subtrees up to isomorphism, compare [AMV06b], Example 3.8. We call these trees the *rational* (or *regular*) CCS processes modulo strong bisimilarity.

Lemma 4.50. *The union of all images of the solutions of finitary flat equation morphisms with parameters from C_{ra} in the initial CIA form precisely the set of all rational trees, i. e.*

$$\bigcup_{\substack{e:X \to HX + C_{\mathrm{ra}} \\ X \text{ finitely presentable}}} e^\dagger[X] = C_{\mathrm{ra}} \,.$$

Proof. We already mentioned that the initial iterative algebra is carried by C_{ra}. The rest of the proof uses Remark 4.39(1) and is analogous to the second and third part of the proof of Theorem 4.41 for streams. □

Adding finite summation $+$ and prefixing $a.-$ does not enlarge the set of trees which occur in the images of solutions as we show next. To this end, let ℓ be the restriction of the abstract GSOS rule defined below (4.10) in Section 4.2.3 to the coproduct components $(\coprod_{n\in\mathbb{N}} X^n) + A \times X$ of KX corresponding to finite summation and prefixing.

Theorem 4.51. *The union of all images of the solutions of finitary flat equation morphisms $e : X \to HFX + C_{\mathrm{ra}}$ in the CIA (C, k) (using the operations $+$ and $a.-$ defined by ℓ) form precisely the set of all rational trees, i. e.*

$$\bigcup_{\substack{e:X\to HFX+C_{\mathrm{ra}}\\ X \text{ finitely presentable}}} e^\dagger[X] = C_{\mathrm{ra}}\,.$$

Proof. We prove the desired equality of sets by considering the two subset inclusions separately:

$\subseteq$ Every finitary flat equation morphism $e : X \to \mathcal{P}_{\mathrm{f}}(A \times FX) + C_{\mathrm{ra}}$ with solution $e^\dagger$ in the CIA (C, k) can be reorganized into one of the form $\bar{e} : X + Y \to \mathcal{P}_{\mathrm{f}}(A \times (X + Y)) + C_{\mathrm{ra}}$ with solution $\bar{e}^\dagger$ in the CIA (C, c^{-1}) such that $\bar{e}^\dagger \cdot \mathsf{inl} = e^\dagger$. This is due to the fact that the operation c^{-1} is nothing but summation of prefixed agents. Then the desired subset inclusion follows from Lemma 4.50.

$\supseteq$ This inclusion is trivial, since for every rational tree $t \in C_{\mathrm{ra}}$ we have the finitary flat equation morphism $e : X \to HFX + C_{\mathrm{ra}}$ given by $X = \{x\}$ and $e(x) = t$. Clearly we have $e^\dagger[X] = \{t\}$, which concludes the proof. □

Agents defined by finitary flat equation morphisms e using the operations $+$ and $a.-$ are precisely the "finite state agents" from Chapter 7.5 of [Mil89].

However, adding parallel composition $|$ results in non-rational trees as solutions, even if we only consider the operations c^{-1} and $|$, as the following example shows.

Example 4.52. Consider the system of recursive equations given by $e(x) = a.y$ and $e(y) = b.(x|x)$. It has a unique solution $e^\dagger$ according to Theorem 4.29, and we shall denote $e^\dagger(x)$ and $e^\dagger(y)$ also by $x^\dagger$ and $y^\dagger$. Let us further denote the tree $\underbrace{x^\dagger|\ldots|x^\dagger}_{n \text{ times}}$ by $(x^\dagger)^n$.

We first establish for all $n \geq 1$ that $(x^\dagger)^n$ has $(x^\dagger)^{n+1}$ as a subtree. To this end, use $x^\dagger = a.y^\dagger = a.b.(x^\dagger|x^\dagger)$ which is immediate from the system e to see that $(x^\dagger)^n = a.b.(x^\dagger|x^\dagger)|(x^\dagger)^{n-1}$. Now by two applications of the first rule for parallel composition $|$ from (4.10) we see that from the root of the tree $(x^\dagger)^n$ there is a path of length 2 labeled by a and then b to the subtree $(x^\dagger|x^\dagger)|(x^\dagger)^{n-1} = (x^\dagger)^{n+1}$. We conclude that $x^\dagger$ has all trees $(x^\dagger)^n$, $n \geq 1$, as

subtrees.
Second, we see that all the above trees are different as strongly extensional trees. It suffices to exhibit for all $n \geq 1$ a path (starting from the root) in the tree $(x^\dagger)^{n+1}$ which is not present in the trees $(x^\dagger)^i$, $1 \leq i \leq n$. Using $(x^\dagger)^{n+1} = (a.b.(x^\dagger|x^\dagger))^{n+1}$ and the rules for $|$ from (4.10) ($n+1$ times), we find a path in $(x^\dagger)^{n+1}$ starting with $n+1$ edges labeled by a. But such a path cannot exist in $(x^\dagger)^i = (a.b.(x^\dagger|x^\dagger))^i$, $1 \leq i \leq n$, which is again due to the rules for $|$ from (4.10) applied i times; for the only path of length i where all edges are labeled by a the $(i+1)$st application of the rules shows that every following edge must be labeled by b.

4.3.3 Regular and Context-Free Languages

Recall from Section 4.2.4 that coalgebras for the functor $HX = X^A \times 2$ are the (possibly infinite) deterministic automata with (finite) input alphabet A. Further recall that the initial CIA for this functor is carried by $C = \mathcal{P}(A^*)$, the set of all formal languages over A.

Regular Languages

Restricting to finite sets X, the coalgebras for $HX = X^A \times 2$ are precisely the deterministic finite automata with input alphabet A. Let us denote the set of all regular languages by C_{reg}.

Lemma 4.53 ([AMV06b], Example 3.7)**.** *The union of all images of the solutions of finitary flat equation morphisms with parameters from C_{reg} in the initial CIA form precisely the set of all regular languages, i. e.*

$$\bigcup_{\substack{e:X\to HX+C_{\mathrm{reg}} \\ X \text{ finitely presentable}}} e^\dagger[X] = C_{\mathrm{reg}}\,.$$

Proof. Analogously to our argumentation for streams from $\mathbb{R}^\omega_{\mathrm{ep}}$ in the proof of Theorem 4.41, this follows from Remarks 4.39(1) and (2): formal languages L can be viewed as Σ-trees for the functor $HX = X^A \times 2$, which are infinite $|A|$-branching trees where every node is labeled in 2; the correspondence is given by $w \in L$ if and only if the path in the tree starting at the root and labeled by the letters of w ends in a node labeled by 1. And the rational such trees correspond precisely to the regular languages, which can be characterized by having only finitely many different derivatives (cf. Remark 5.43 below). □

Next we show that adding the empty language $\emptyset$, the empty-word language $\{\varepsilon\}$, the single-letter language $\{a\}$ for every $a \in A$, and the operations

union $\cup$, intersection $\cap$, complement $\overline{(-)}$ and prefixing $a.-$ for every $a \in A$ (where $a.L = \{aw \mid w \in L\}$), does not increase expressiveness of finite recursive specifications. To this end, we define them via an abstract GSOS rule $\ell : K(H \times \mathrm{Id}) \to HF$ where $KX = 1+1+A+X\times X+X\times X+X+A\times X$. According to Remark 3.18 it suffices to specify $\ell' : KH \to HF$; for every set X, the seven coproduct components of ℓ'_X are given by the following assignments:

$$\begin{aligned}
\emptyset &\mapsto ((\emptyset)_{a\in A}, 0)\\
\varepsilon &\mapsto ((\emptyset)_{a\in A}, 1)\\
b &\mapsto ((t_a), 0) \quad \text{where } t_a = \begin{cases} \varepsilon & a = b \\ \emptyset & a \neq b \end{cases}\\
((x_a), j) \cup ((y_a), k) &\mapsto ((x_a \cup y_a), j \vee k)\\
((x_a), j) \cap ((y_a), k) &\mapsto ((x_a \cap y_a), j \wedge k)\\
\overline{((x_a), j)} &\mapsto ((\overline{x_a}), \neg j)\\
b.((x_a), j) &\mapsto ((t_a), 0)\\
&\text{where } t_a = \begin{cases} \bigcup_{a\in A} a.x_a & a = b \text{ and } j = 0 \\ (\bigcup_{a\in A} a.x_a) \cup \varepsilon & a = b \text{ and } j = 1 \\ \emptyset & a \neq b \end{cases}
\end{aligned}$$

For the first, second and fourth component, compare the first three assignments of (4.11) in Section 4.2.4, and for the fifth one compare the initial example from Section 4.2.4. It is not difficult to verify that the coproduct components of the ℓ-interpretation $b : KC \to C$ are indeed the desired operations. We show that every finitary flat equation morphism $e : X \to HFX + C_{\mathrm{reg}}$ can be translated into a finitary flat equation morphism $\bar{e} : X + Y \to H(X + Y) + C_{\mathrm{reg}}$ such that $\bar{e}^\dagger \cdot \mathsf{inl} = e^\dagger$ where $e^\dagger$ is the unique solution of e in $k = c^{-1} \cdot H\widehat{b}$ (see Theorem 4.7) and $\bar{e}^\dagger$ is the unique solution of $\bar{e}$ in c^{-1}.

Construction 4.54. Given a finitary flat equation morphism $e : X \to HFX + C_{\mathrm{reg}}$, we initially set $Y = \emptyset$ and perform the following steps:

1. For the empty language, the empty-word language and every single-letter languages we add a variable to Y, i. e. we set $Y = \{y_\emptyset, y_\varepsilon\} \cup \{y_a \mid a \in A\}$. We modify e to e_1 by replacing all occurrences of the languages $\emptyset$, ε and a, $a \in A$, by their variables and by adding one equation for

every language as follows:

$$y_\emptyset = ((y_\emptyset)_{a\in A}, 0)$$
$$y_\varepsilon = ((y_\emptyset)_{a\in A}, 1)$$
$$y_b = ((t_a), 0) \quad \text{for every } b \in A \text{ where } t_a = \begin{cases} y_\varepsilon & a = b \\ y_\emptyset & a \neq b\,. \end{cases}$$

2. For every occurrence of the prefixing operation $b.t$ on the right-hand side of equations we add a variable y to Y. We modify e_1 to e_2 by replacing the subterm $b.t$ by y and by adding the equation

$$y = ((t_a), 0) \quad \text{where } t_a = \begin{cases} t & a = b \\ y_\emptyset & a \neq b\,. \end{cases}$$

3. Observe that—due to the previous steps—on the right-hand sides of equations from e_2 only the operations $\cup$, $\cap$ and $\overline{(-)}$ can occur. Now we repeat the following steps:

   (a) Analogously to formulas from propositional logic with logical connectives $\wedge$, $\vee$ and $\neg$ we can rewrite all the right-hand side terms which are not single variables into a conjunctive normal form (CNF) with $|X + Y|$ literals per clause.

   (b) For every CNF γ for which no variable was introduced already in previous runs of this loop, we add a new variable y_γ to Y and add the equation $y_\gamma = \gamma$. If there is no such CNF, we exit the loop.

   (c) Since the new right-hand sides γ from step (3b) are not elements of $H(X+Y)$, for every such γ we replace all variables by the right-hand sides of their equations (which are elements of $H(X + Y)$). We use the components for $\cup$, $\cap$ and $\overline{(-)}$ of the above definition of ℓ'_{X+Y} to distribute these three operations over the operation c^{-1} represented by H in order to obtain the format $HF(X + Y)$. Since again the operations $\cup$, $\cap$ and $\overline{(-)}$ occur on the right-hand sides of the newly added equations, we repeat the loop.

4. When the loop has finished, for every CNF γ we replace all its occurrences by the corresponding variable y_γ. The resulting equation morphism is $\bar{e} : X + Y \to H(X + Y) + C_{\mathrm{reg}}$.

Lemma 4.55. *For every finitary flat equation morphism $e : X \to HFX + C_{\mathrm{reg}}$ with solution $e^\dagger$ in the CIA (C, k), Construction 4.54 terminates and outputs a finitary flat equation morphism $\bar{e} : X + Y \to H(X + Y) + C_{\mathrm{reg}}$ with solution $\bar{e}^\dagger$ in the CIA (C, c^{-1}) such that $\bar{e}^\dagger \cdot \mathsf{inl} = e^\dagger$.*

Proof. First, we see that Construction 4.54 terminates after finitely many steps and yields a finitary flat equation morphism $\bar{e}$: in step (1) $|A| + 2$ variables and equations are added, and A is a finite alphabet. For step (2) observe that in finitely many equations with finite terms on the right-hand sides there can only exist finitely many occurrences of the prefixing operation, and this is precisely the number of variables and equations that are added. Finally, the loop in step (3) terminates since there are only finitely many CNFs over finitely many variables, and at most that many variables and equations are added.

Second, Construction 4.54 applied to e indeed results in $\bar{e}$ such that $\bar{e}^\dagger \cdot \mathsf{inl} = e^\dagger$: all changes to the system of recursive equations in the construction preserve the solutions of the original variables from X in the CIA k. For substitution of a new variable y for a right-hand side term t and adding the equation $y = t$ this is clear for solutions in any algebra. For the variables added in step (1) it is easy to see that that they are solved to the intended languages. For step (2) observe that prefixing is a special case of the operation c^{-1} and nothing but the corresponding embedding is performed in this step. Forming the CNFs in step (3a) uses laws for the operations $\cup$, $\cap$ and $\overline{(-)}$ that are well-known, and the distribution of these operations over c^{-1} in step (3c) is done according to the defining laws for the three operations encoded in ℓ'_{X+Y}. Finally, a solution in the CIA k where the additional operations $b : FC \to C$ are not used, reduces to a solution in the CIA c^{-1}. □

Theorem 4.56. *The union of all images of the solutions of finitary flat equation morphisms $e : X \to HFX + C_{\mathrm{reg}}$ in the CIA (C, k) (using the seven operations defined by ℓ) form precisely the set of all regular languages, i. e.*

$$\bigcup_{\substack{e:X\to HFX+C_{\mathrm{reg}} \\ X \text{ finitely presentable}}} e^\dagger[X] = C_{\mathrm{reg}} \,.$$

Proof. We prove the desired equality of sets by considering the two subset inclusions separately:

$\subseteq$ We apply Construction 4.54 and obtain for each e a finitary flat equation morphism $\bar{e} : X + Y \to H(X + Y) + C_{\mathrm{reg}}$. We know from Lemma 4.53 that the images of the solutions of the equation morphisms $\bar{e} : X + Y \to H(X + Y) + C_{\mathrm{reg}}$ are subsets of C_{reg}. Since we have $\bar{e}^\dagger \cdot \mathsf{inl} = e^\dagger$ by Lemma 4.55, we conclude that also the images of the solutions of the equation morphisms e are subsets of C_{reg}.

$\supseteq$ This inclusion is trivial, since for every regular language $L \in C_{\mathrm{reg}}$ we have the finitary flat equation morphism $e : X \to HFX + C_{\mathrm{reg}}$ given by $X = \{x\}$ and $e(x) = L$. Clearly we have $e^\dagger[X] = \{L\}$, which concludes the proof. □

Context-Free Languages

We have already seen in Construction 4.32 of Section 4.2.4 that context-free grammars in Greibach normal form can be translated to finitary flat equation morphisms (using the empty language $\emptyset$, the empty-word language $\{\varepsilon\}$ and the operations of union $\cup$ and concatenation $\cdot$). It is not difficult to see that these concepts essentially correspond to each other. Thus, defining the four operations $\emptyset$, ε, $\cup$ and $\cdot$ via the abstract GSOS rule $\bar{\ell}$ given by (4.11), and denoting the set of all context-free languages over A by C_{cf}, we see that the solutions of finitary flat equation morphisms $e : X \to H\bar{F}X + C_{\mathrm{cf}}$ in the CIA $(C, \bar{k})$ (using the four operations) assign to every variable from X an element of C_{cf}.

The following example shows that using concatenation alone suffices to obtain a context-free language as a solution of a variable of a finitary flat equation morphism.

Example 4.57. Consider the alphabet $A = \{a_1, a_2\}$. The finitary flat equation morphism e given by the system

$$\begin{aligned} x &= ((t_a), 1) \quad \text{where } t_{a_1} = x \cdot a_2 \text{ and } t_{a_2} = y \\ y &= ((y)_{a\in A}, 0) \end{aligned}$$

of recursive equations has the unique solution $e^\dagger$ given by $e^\dagger(x) = \{a_1^n a_2^n \mid n \in \mathbb{N}\}$ and $e^\dagger(y) = \emptyset$. And $\{a_1^n a_2^n \mid n \in \mathbb{N}\}$ is not regular but context-free.

However, notice that to define and use concatenation in (flat) equation morphisms, we need union anyway, see the last equation in (4.11). Concatenation, in turn, can be used to define the Kleene star operation $(-)^*$ by the following assignment (as part of a GSOS rule $\ell : K(H \times \mathrm{Id}) \to HF$):

$$((x_a), j, x)^* \mapsto ((t_a), 1) \quad \text{where } t_a = x_a \cdot x^*$$

Now let $\ell : K(H \times \mathrm{Id}) \to HF$ define the languages $\emptyset$, $\{\varepsilon\}$ and $\{a\}$ for every $a \in A$, and the operations $\cup$, $a.-$ for every $a \in A$, $\cdot$ and $(-)^*$; i. e. we have $KX = 1{+}1{+}A{+}X{\times}X{+}A{\times}X{+}X{\times}X{+}X$. Similar to Construction 4.54, the following translation of finitary flat equation morphisms $e : X \to HFX + C_{\mathrm{cf}}$ to finitary flat equation morphisms $\bar{e} : X + Y \to H\bar{F}(X + Y) + C_{\mathrm{cf}}$ ($\bar{F}$ builds terms using only $\emptyset$, ε, $\cup$ and $\cdot$) shows that this choice of operations guaranties context-free solutions.

Construction 4.58. Given a finitary flat equation morphism $e : X \to HFX + C_{\mathrm{cf}}$, we initially set $Y = \emptyset$ and perform the following steps:

1. See step (1) of Construction 4.54 and obtain e_1 from e.

2. See step (2) of Construction 4.54 and obtain e_2 from e_1.

3. Observe that—due to the previous steps—the only operation on the right-hand sides of equations from e_2 that needs to be eliminated is $(-)^*$. To this end, we repeat the following steps until all occurrences of $(-)^*$ are eliminated:

   (a) We pick a term t^* where t is a term that only uses union and concatenation.

   (b) We add a new variable y to Y and add the equation $y = t^*$.

   (c) Since the new right-hand side t^* from step (3b) is no element of $H\bar{F}(X + Y)$, we replace all variables by the right-hand sides of their equations (which are elements of $HF(X + Y)$). We use the components for $\cup$, $\cdot$ and finally $(-)^*$ of the above ℓ_{X+Y} to distribute these three operations over the operation c^{-1} represented by H in order to obtain the format $HF(X + Y)$.

   (d) We replace all occurrences of t^* by y and start the next run of the loop if there is still $(-)^*$ in the system of recursive equations.

4. The resulting equation morphism is $\bar{e} : X + Y \to H\bar{F}(X + Y) + C_{\mathrm{cf}}$.

Recall the CIA $(C, \bar{k})$ mentioned above Example 4.57.

Lemma 4.59. *For every finitary flat equation morphism $e : X \to HFX+C_{\mathrm{cf}}$ with solution $e^\dagger$ in the CIA (C, k), Construction 4.54 terminates and outputs a finitary flat equation morphism $\bar{e} : X+Y \to H\bar{F}(X+Y)+C_{\mathrm{cf}}$ with solution $\bar{e}^\dagger$ in the CIA $(C, \bar{k})$ such that $\bar{e}^\dagger \cdot \mathsf{inl} = e^\dagger$.*

Proof. First, we see that Construction 4.58 terminates after finitely many steps and yields a finitary flat equation morphism $\bar{e}$: for step (1) and (2) see the analysis for Construction 4.54 in the proof of Lemma 4.55 above. The loop in step (3) terminates since every run removes one of the finitely many occurrences of $(-)^*$ (and does not add a new one since the only occurrences that are added in step (3c) have the form t^* according to the $(-)^*$-component of ℓ_{X+Y} and are immediately removed in step (3d)). In every run of the loop precisely one new variable and equation is added.

Second, Construction 4.58 applied to e results in $\bar{e}$ such that $\bar{e}^\dagger \cdot \mathsf{inl} = e^\dagger$: again we rely on the analysis of Construction 4.54 in the proof of Lemma 4.55 for steps (1) and (2). The distribution of $\cup$, $\cdot$ and $(-)^*$ over c^{-1} in step (3c) is done according to the defining laws for the three operations encoded in ℓ_{X+Y}. Finally, a solution in the CIA (C, k) where the operations a for every $a \in A$, $a.-$ for every $a \in A$ and $(-)^*$ are not used, reduces to a solution in the CIA $(C, \bar{k})$. □

Theorem 4.60. *The union of all images of the solutions of finitary flat equation morphisms $e : X \to HFX + C_{\mathrm{cf}}$ in the CIA (C, k) (using the seven operations defined by ℓ) form precisely the set of all context-free languages, i. e.*

$$\bigcup_{\substack{e:X\to HFX+C_{\mathrm{cf}} \\ X \text{ finitely presentable}}} e^\dagger[X] = C_{\mathrm{cf}} \,.$$

Proof. We prove the desired equality of sets by considering the two subset inclusions separately:

$\subseteq$ We apply Construction 4.58 and obtain for each e a finitary flat equation morphism $\bar{e} : X + Y \to H\bar{F}(X + Y) + C_{\mathrm{cf}}$. We know from our above argumentation (the first paragraph of the context-free languages section) that the images of the solutions of the equation morphisms $\bar{e} : X + Y \to H\bar{F}(X+Y)+C_{\mathrm{cf}}$ are subsets of C_{cf}. Since we have $\bar{e}^\dagger \cdot \mathsf{inl} = e^\dagger$ by Lemma 4.59, we conclude that also the images of the solutions of the equation morphisms e are subsets of C_{cf}.

$\supseteq$ This inclusion is trivial, since for every context-free language $L \in C_{\mathrm{cf}}$ we have the finitary flat equation morphism $e : X \to HFX + C_{\mathrm{cf}}$ given by $X = \{x\}$ and $e(x) = L$. Clearly we have $e^\dagger[X] = \{L\}$, which concludes the proof. □

Non-Context-Free Languages

Since context-free languages are not closed under intersection and complement, we clearly obtain non-context-free languages as solutions of finitary flat equation morphisms which use concatenation and intersection/complement:

Example 4.61. Consider the alphabet $A = \{a_1, a_2, a_3\}$ and the finitary flat equation morphism e given by the left-hand system

$$\begin{aligned}
x &= (((y_1 \cdot a_2 \cdot y_2) \cap (z_1 \cdot z_2), \emptyset, \emptyset), 1) & e^\dagger(x) &= \{a_1^n a_2^n a_3^n \mid n \in \mathbb{N}\} \\
y_1 &= ((y_1 \cdot a_2, \emptyset, \emptyset), 1) & e^\dagger(y_1) &= \{a_1^n a_2^n \mid n \in \mathbb{N}\} \\
y_2 &= ((\emptyset, \emptyset, y_2), 1) & e^\dagger(y_2) &= c^* \\
z_1 &= ((z_1, \emptyset, \emptyset), 1) & e^\dagger(z_1) &= a^* \\
z_2 &= ((\emptyset, z_2 \cdot a_3, \emptyset), 1) & e^\dagger(z_2) &= \{a_2^n a_3^n \mid n \in \mathbb{N}\}
\end{aligned}$$

of recursive equations where the families of terms $(t_a)_{a\in A}$ are written as triples $(t_{a_1}, t_{a_2}, t_{a_3})$ and where in particular concatenation and intersection are used. Its unique solution $e^\dagger$ is shown on the right-hand side; as we see, the solution for x is not a context-free, but a recursive language.

This immediately gives an example for complement, too, since we could have written $\overline{\overline{y_1 \cdot a_2 \cdot y_2} \cup \overline{z_1 \cdot z_2}}$ instead of the intersection but with the same resulting solution for x.

4.4 Operations Beyond Abstract GSOS

Arbitrary Monads

Recall from Lemma 3.14 that every abstract GSOS rule $\ell : K(H \times \mathrm{Id}) \to HF$ corresponds to a distributive law λ of the monad F over the cofree copointed functor $H \times \mathrm{Id}$. But there are also distributive laws of other monads than those free ones over $H \times \mathrm{Id}$.

Indeed, Theorems 4.7 and 4.8 hold more generally for an arbitrary monad M in lieu of the free one F. More detailed, recall from Construction 3.13 (and the sentence below it) that every distributive law λ of a monad M over the cofree copointed functor $H \times \mathrm{Id}$ induces a λ-interpretation, i. e., a unique Eilenberg-Moore algebra $b : MC \to C$ such that

$$\langle c, \mathrm{id}_C \rangle \cdot b = (H \times \mathrm{Id})b \cdot \lambda_C \cdot M\langle c, \mathrm{id}_C \rangle \,.$$

The version of Theorem 4.3 presented in [Bar03, Bar04] and dually in [UVP01] states that for every $e : X \to HMX$ there is a unique solution $e^\dagger : X \to C$, i. e., $e^\dagger$ is such that (4.4) commutes with M in lieu of F and $b : MC \to C$ in lieu of $\widehat{b} : FC \to C$. Then Theorem 4.7 shows that (C, k) is a CIA for HM. Similarly, Theorem 4.5 clearly holds for an arbitrary monad M. Thus, Theorem 4.8 shows that (C, k') is a CIA for MHM. Finally, the development of Section 4.1.3 is independent of F, i. e. provided HM and MHM are iteratable, Corollaries 4.13 and 4.14 hold with F replaced by M. The compositionality properties of Remarks 4.9 and 4.15 and the overview of formats given at the end of Section 4.1.3 remain valid if every occurrence of F is replaced by the monad M.

Pointed Functors

Even more generally, observe that our proof of Theorem 4.7 only makes use of the unit $\eta : \mathrm{Id} \to M$ of the monad M. In fact, there are versions of Theorems 4.7 and 4.8 that hold for a pointed functor M in lieu of a monad and for a given distributive law of M over the cofree copointed functor $H \times \mathrm{Id}$ or the functor H, respectively. However, in this case we need to assume that our base category is cocomplete:

Assumption 4.62. We now assume that our base category is cocomplete. We furthermore assume that (M, η) is a pointed functor and H is (as before) an endofunctor with a final coalgebra (C, c).

Remarks 4.63. 1. Every distributive law $\lambda : MH \to HM$ gives one of the cofree copointed functor $H \times \mathrm{Id}$ via

$$\begin{array}{ccc} MH & \xrightarrow{\lambda} & HM \\ {\scriptstyle M\pi_0}\uparrow & & \uparrow{\scriptstyle \pi_0} \\ M(H \times \mathrm{Id}) & \dashrightarrow & (H \times \mathrm{Id})M \\ & {\scriptstyle M\pi_1}\searrow & \downarrow{\scriptstyle \pi_1} \\ & & M \end{array} \tag{4.13}$$

but not conversely.

2. Recall from Construction 3.13 (and the sentence below it) that we have for any distributive law λ of M over the cofree copointed functor $H \times \mathrm{Id}$ a unique λ-*interpretation*, i. e., a unique morphism $b : MC \to C$ such that the diagram below commutes

$$\begin{array}{ccccc} MC & \xrightarrow{M\langle c, \mathrm{id}_C\rangle} & M(HC \times C) & \xrightarrow{\lambda_C} & (H \times \mathrm{Id})MC \\ {\scriptstyle b}\downarrow & & & & \downarrow{\scriptstyle (H\times\mathrm{Id})b} \\ C & & \xrightarrow{\langle c, \mathrm{id}_C\rangle} & & HC \times C \end{array}$$

and (C, b) is an algebra for the pointed functor M. Notice that $b : MC \to C$ here corresponds to $\widehat{b} : FC \to C$ in Theorem 3.17. In case the distributive law arises from a distributive law λ of M over H as in (4.13), the above diagram simplifies to

$$\begin{array}{ccccc} MC & \xrightarrow{Mc} & MHC & \xrightarrow{\lambda_C} & HMC \\ {\scriptstyle b}\downarrow & & & & \downarrow{\scriptstyle Hb} \\ C & & \xrightarrow{c} & & HC\,. \end{array} \tag{4.14}$$

So $b : MC \to C$ is both the λ-interpretation and the interpretation of the extension of λ according to (1).

Next we shall need a version of Theorem 4.3 for a given distributive law λ of M over H (or over the cofree copointed functor $H \times \mathrm{Id}$). This is a variation of Theorem 4.2.2 of Bartels [Bar04] (see also Lemma 4.3.2 in loc. cit.) using the cocompleteness of our base category.

Theorem 4.64 (cf. [Bar04])**.** *Let $\lambda : MH \to HM$ be a distributive law of the pointed functor M over the functor H, and let $b : MC \to C$ be the λ-interpretation. Then for every λ-*equation *$e : X \to HMX$ there exists a unique solution, i. e., a unique morphism $e^\dagger : X \to C$ such that the diagram below commutes:*

$$\begin{array}{ccc} X & \xrightarrow{e^\dagger} & C \\ {\scriptstyle e}\downarrow & & \downarrow{\scriptstyle c} \\ & & HC \\ & & \uparrow{\scriptstyle Hb} \\ HMX & \xrightarrow{HMe^\dagger} & HMC \end{array} \tag{4.15}$$

Since one part of the proof in [Bar04] (the uniqueness part) is only presented for Set, we give a full proof in Appendix A for the convenience of the reader.

Remark 4.65. As explained by Bartels in [Bar04], Theorem 4.64 extends to the case where a distributive law λ of M over the cofree copointed functor $H \times \mathrm{Id}$ is given. We briefly explain the ideas.

Let $D = H \times \mathrm{Id}$ and $\varepsilon = \pi_1 : D \to \mathrm{Id}$.

1. It follows from the proof of Lemma 3.9 that

   $$C \xrightarrow{\langle c, \mathrm{id}_C \rangle} HC \times C$$

   is the final coalgebra for the cofree copointed functor (D, ε).

2. One verifies that λ-equations $e : X \to HMX$ are in bijective correspondence with morphisms $f : X \to DMX$ such that

   $$\begin{array}{ccc} X & \xrightarrow{f} & DMX \\ & {\scriptstyle \eta_X}\searrow & \downarrow{\scriptstyle \varepsilon_{MX}} \\ & & MX \end{array}$$

   and that solutions of e correspond bijectively to solutions of f, i. e., morphisms $f^\dagger : X \to C$ such that diagram (4.15) commutes with H replaced by D and c replaced by $\langle c, \mathrm{id}_C \rangle$:

   $$\begin{array}{ccc} X & \xrightarrow{f^\dagger} & C \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle \langle c, \mathrm{id}_C \rangle} \\ & & DC \\ & & \uparrow{\scriptstyle Db} \\ DMX & \xrightarrow{DMf^\dagger} & DMC \end{array}$$

See [Bar04], Lemma 4.3.9.

3. The same proof as the one for Theorem 4.64 shows that for every $f : X \to DMX$ as in (2) above there exists a unique solution $f^\dagger$. One only replaces H by D, c by $\langle c, \mathrm{id}_C \rangle$, and one has to verify that the D-coalgebra $\bar{e} : SX \to DSX$ from (A.1) is a coalgebra for the copointed endofunctor (D, ε), see [Bar04], Lemma 4.3.7.

To sum up, we obtain the following

Corollary 4.66 (cf. [Bar04])**.** *Let λ be a distributive law of the pointed functor M over the copointed one $H \times \mathrm{Id}$, and let $b : MC \to C$ be the λ-interpretation. Then for every $e : X \to HMX$ there exists a unique solution, i. e., a unique $e^\dagger : X \to C$ such that* (4.15) *commutes.*

Theorem 4.67. (Unique solutions of flat equation morphisms $X \to HMX + C$)**.** *Let λ be a distributive law of the pointed functor M over the copointed one $H \times \mathrm{Id}$, and let $b : MC \to C$ be the λ-interpretation. Consider the HM-algebra*

$$k = (\ HMC \xrightarrow{Hb} HC \xrightarrow{c^{-1}} C\ )\ .$$

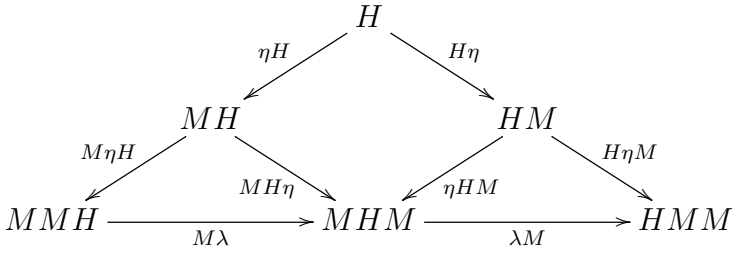
Then (C, k) is a CIA for HM.

Indeed, to prove this result copy the proof of Theorem 4.7 replacing F by M and $\widehat{b} : FC \to C$ by $b : MC \to C$.

However, for our version of Theorem 4.8 in the current setting we need a different proof. Also, we need to restrict ourselves to distributive laws of the pointed functor M over the functor H. We start with an auxiliary lemma.

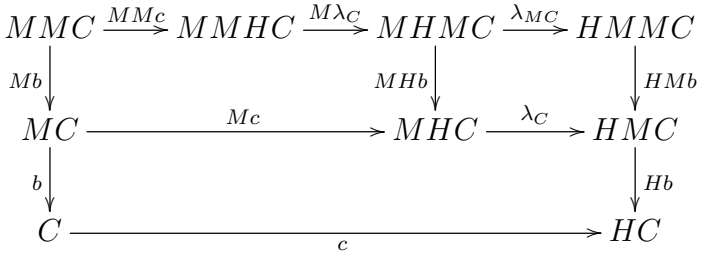
Lemma 4.68. *Let $\lambda : MH \to HM$ be a distributive law of the pointed functor M over the functor H, and let $b : MC \to C$ be the λ-interpretation. Then the natural transformation $\lambda' = \lambda M \cdot M\lambda : MMH \to HMM$ is a distributive law of the pointed functor MM over H, and $Mb \cdot b : MMC \to C$ is the λ'-interpretation in C.*

Proof. Clearly $(MM, \eta M \cdot \eta = M\eta \cdot \eta : \mathrm{Id} \to MM)$ is a pointed endofunctor. The following commutative diagram



shows that $\lambda' = \lambda M \cdot M\lambda$ is a distributive law of the pointed functor MM over H. In fact, the triangles commute by the assumption on λ, and the remaining upper square commutes by naturality of η; thus the outside triangle commutes.

To see that $b \cdot Mb$ is the λ'-interpretation in C, consider the following diagram:



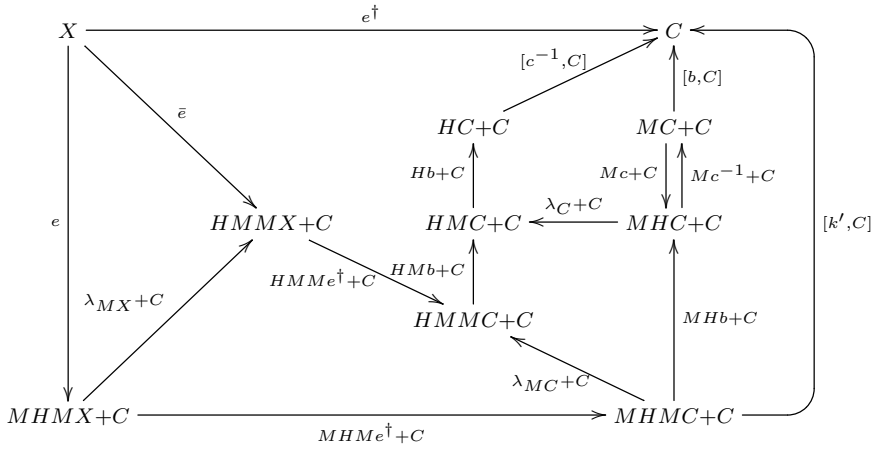
It commutes since $b : MC \to C$ is the λ-interpretation in C and by the naturality of λ. This concludes the proof. □

Theorem 4.69. (Unique solutions of flat equation morphisms $X \to MHMX + C$)**.** *Let $\lambda : MH \to HM$ be a distributive law of the pointed functor M over the functor H, and let $b : MC \to C$ be the λ-interpretation. Consider the MHM-algebra*

$$k' = (MHMC \xrightarrow{Mk} MC \xrightarrow{b} C)\,,$$

where $k = c^{-1} \cdot Hb$. Then (C, k') is a CIA for MHM.

Proof. We have to prove that for every flat equation morphism $e : X \to MHMX + C$ for MHM there is a unique solution $e^\dagger : X \to C$ in $k' = b \cdot Mc^{-1} \cdot MHb : MHMC \to C$, i. e., a unique morphism $e^\dagger$ such that the outside of the diagram



commutes. To this end, we define the flat equation morphism

$$\bar{e} = (X \xrightarrow{e} MHMX + C \xrightarrow{\lambda_{MX}+C} HMMX + C)$$

for HMM (this is the left-hand triangle). According to Lemma 4.68 and Theorem 4.67 (applied to the extension of λ from Remark 4.63),

$$HMMC \xrightarrow{HMb} HMC \xrightarrow{Hb} HC \xrightarrow{c^{-1}} C$$

is a CIA for HMM. So $\bar{e}$ has a unique solution $e^\dagger$ in this CIA, i. e., the big inner part of the diagram commutes. In the upper right-hand part, $b : MC \to C$ is the λ-interpretation in C. Since that part and the two remaining squares also commute (due to naturality of λ), the desired outside commutes. Thus, $e^\dagger$ also is a solution of e in the algebra $k' : MHMC \to C$.

This solution is unique, since any other solution s of e in k' (i. e., the outside of the diagram with s in lieu of $e^\dagger$ commutes) is a solution of $\bar{e}$ in the CIA $c^{-1} \cdot Hb \cdot HMb : HMMC \to C$ (i. e., the inner part commutes with s in lieu of $e^\dagger$), thus $s = e^\dagger$. □

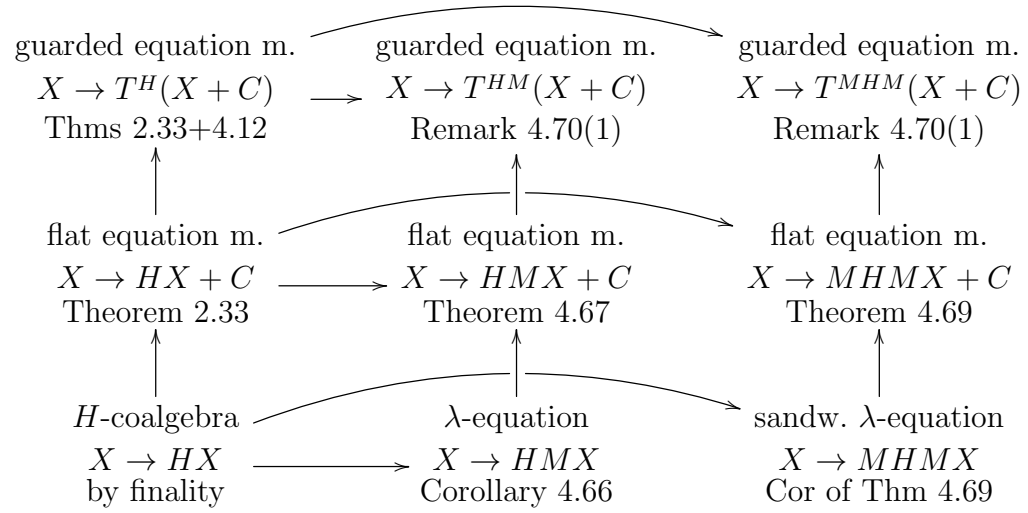
We do not see how to prove Theorem 4.69 for a distributive law λ of M over the cofree copointed functor $H \times \mathrm{Id}$: in this case we cannot form an equation morphism $\bar{e}$ as in the proof of Theorem 4.69, and also a proof via a result analogous to Theorem 4.5 is not possible since the monad multiplication is required in the proof.

Remarks 4.70. 1. Since the extension to infinite terms in Section 4.1.3 works for every CIA, we clearly can apply it to the CIAs from Theorems 4.67 and 4.69. We obtain Corollaries analogous to Corollaries 4.13 and 4.14 which rely on distributive laws $\lambda : M(H \times \mathrm{Id}) \to (H \times \mathrm{Id})M$ and $\lambda : MH \to HM$ of a pointed functor M over the copointed functor $H \times \mathrm{Id}$ and the functor H, respectively, instead of an abstract GSOS rule $\ell : K(H \times \mathrm{Id}) \to HF$.

2. Also, both compositionality principles from Remarks 4.9 and 4.15 can be applied here since they hold true for every CIA.

We summarize and complete the results of this section in the following

overview.



This overview is similar as the one at the end of Section 4.1.3; the leftmost column is even identical since it is independent of a GSOS rule or distributive law. The difference for the middle and right-hand columns is that instead of a GSOS rule $\ell : K(H \times \mathrm{Id}) \to HF$ distributive laws $\lambda : M(H \times \mathrm{Id}) \to (H \times \mathrm{Id})M$ (middle column) or $\lambda : MH \to HM$ (right-hand column) of a pointed functor M over the cofree copointed functor $H \times \mathrm{Id}$ or the functor H are the underlying structure. The different kinds of distributive laws for the middle and right-hand columns imply that the formats in the right-hand column do no longer generalize the ones in the middle column. Finally keep in mind that all formats shown in the middle and right-hand columns of this overview require the base category to be cocomplete, cf. Assumption 4.62.

4.5 Related Work

Connections between recursive equations and distributive laws have been investigated earlier. Klin [Kli04] worked in a CPO-enriched setting in order to obtain new distributive laws by combining given distributive laws with recursive equations. Jacobs [Jac06b] showed that the coalgebraic view [AMV03] on iterative theories fits into the bialgebraic framework of [Bar04] which uses distributive laws.

ℓ-equations (cf. Definition 4.2 and Theorem 4.3) were investigated by Bartels ([Bar04], Definition 4.2.3) under the name "guarded recursive definition". Independent work in the dual setting can be found in [UVP01] and [CUV06] as explained below Theorem 4.3. However, none of our further results seem to be known or to be the dual of existing results. All facts about CIAs and

(flat) equation morphisms relevant for this thesis are available in Section 2.3 of the preliminaries; for more on this topic, we refer the reader to Milius' paper [Mil05].

Coming to the five examples of Section 4.2, streams have been extensively investigated from a coalgebraic point of view by Rutten, see e. g. [Rut05a, Rut05b]. Later, Silva and Rutten also investigated infinite (binary) trees [SR07, SR10], where they applied similar methods. CCS processes were invented by Milner, see [Mil89] containing many results and examples. For an introduction to formal languages we refer to the textbook [HMU07] by Hopcroft, Motwani and Ullman; the original paper introducing the Greibach normal form for context-free grammars is [Gre65]. A coalgebraic description of context-free grammars in Greibach normal form has recently been given by Winter, Bonsangue and Rutten [WBR11], and previously, a coalgebraic approach to context-free grammars was given by Hasuo and Jacobs [HJ05]. However, those are completely different from our approach. For non-well-founded sets see the books by Aczel [Acz88] and Barwise and Moss [BM96]. The non-well-founded sets examples from Section 4.2.5 were provided by Moss.

The excursion to finite systems of equations in Section 4.3 is the only place where locally finitely presentable categories and finitely presentable objects are needed in our work. Their definitions and several other facts about them can be found in the book [AR94] by Adámek and Rosický. The starting point for iterative algebras is [AMV06b] by Adámek, Milius and Velebil. The subsets of final coalgebras for which we proved definability by finite systems of equations using certain operations have all been known: for eventually periodic or ultimately periodic streams (mentioned in Remark 2.4 of [Rut08]) and rational streams see Rutten [Rut08], for rational or regular CCS processes see Chapter 7.5 of Milner's book [Mil89], and for regular and context-free languages see for example the book [HMU07] by Hopcroft, Motwani and Ullman. A related matter is the question which abstract GSOS rules define operations on these subsets; for the case of rational fixed points [AMV06b] it has been settled in [BMR12] (inspired by [Ace94]), where a rational fixed point for a set functor is the subcoalgebra of the final coalgebra given by the behavior of all finite coalgebras (e. g. eventually periodic streams, rational trees, rational processes or regular languages).

The results in Section 4.4 rest upon Theorem 4.64 which was stated and—for the category Set—proved by Bartels [Bar04]. The proof for an arbitrary cocomplete category given in Appendix A is due to Milius. The extension of Theorem 4.64 to cofree copointed functors stated in Corollary 4.66 was observed in full generality (for all cocomplete categories) by Bartels [Bar04].

The results of Sections 4.1 and 4.4 can also be found in our joint pa-

per [MMS13] with Milius and Moss (the special issue version of the conference paper [MMS10]). However, we provide more details here about compositionality in that we added Remarks 4.15 and 4.70, and about the relation of our formats of recursive equations (Remarks 4.6, 4.10 and 4.16 and Lemmata 4.17 and 4.18 were added). Some of the examples from Section 4.2 are taken from these papers, some are new here. Section 4.3 has not been published previously—only a remark hinting results for formal languages (cf. Section 4.3.3) was present in the papers.

Chapter 5

Compositionality of Recursive Program Schemes

In Chapter 4 we have been concerned with systems of recursive equations which define constants from a final H-coalgebra C. Compositionality of formats for systems of recursive equations has been achieved by allowing parameters from C in the recursive equations. Technically, we formed a coproduct with the object C.

In the present chapter we consider recursive program schemes which define operations on a final H-coalgebra C—one may regard this as stepping one level up from first-order to second-order recursive definitions. We show how compositionality of formats for recursive program schemes can be achieved by using operations newly defined in a scheme (and represented by a functor V) as given operations in another scheme. Technically, we now form a coproduct with the functor V.

In Section 5.1 we first prove compositionality results for the existing format of recursive program schemes. Since certain operations cannot be defined in this format, we second present formats of recursive program schemes which fix this problem and prove compositionality results for them. At the end of the section we give an overview of the different program scheme formats and their relationships. The next section (Section 5.2) provides several examples on the five final coalgebras we have already seen in Section 4.2. These illustrate how concrete operations can be defined via recursive program schemes and how this can be done in a compositional way. Moreover, they show how our theory serves as a roof over several existing formats for the definition of operations on final coalgebras.

For the present chapter, we make the same assumptions as in the previous one (cf. Assumption 4.1).

5.1 Compositional Formats of Recursive Program Schemes

We have already seen at least two formats of recursive program schemes in this thesis which can be used to define operations on a final H-coalgebra: the first one is the ordinary RPS's from Definition 2.52, and the second one is abstract GSOS rules/distributive laws whose interpretations can be considered as solutions, cf. Sections 3.3/3.2. However, in both cases no compositionality results have been known: all operations one wants to define must be defined in one RPS, at least if the definition of operations depends on that of other operations.

In Section 5.1.1 we show for ordinary RPS's that if the given operations form a CIA, the latter is completed by the newly defined operations to a CIA again. Since we can consider this CIA as given operations in another RPS, we obtain an important modularity principle. Moreover, we show how ordinary RPS's can be composed and that modularity extends to a compositionality principle. In Section 5.1.2 we define two formats of recursive program schemes where the given operations are assumed to form the interpretation of an abstract GSOS rule, and show that these given operations are completed by the newly defined operations to an interpretation of an abstract GSOS rule again. Since we can consider this interpretation as given operations in another scheme of either format, we obtain a further modularity principle. Again it extends to a compositionality principle. For these two formats further results are proved concerning the relationship to ordinary RPS's, the relationship to first-order recursive definitions (systems of recursive equations), and the order in which operations are defined.

5.1.1 Ordinary RPS's

Recall from Theorem 2.54 that every guarded recursive program scheme $e : V \to T^{H+V}$ has a unique interpreted solution $e^\ddagger : VA \to A$ in every CIA $a : HA \to A$ for H. We now extend this result by proving that $[a, e^\ddagger] : (H + V)A \to A$ is then a CIA for $(H + V)$.

Theorem 5.1 (Unique solutions of RPS's extend CIAs)**.** *Let $e : V \to T^{H+V}$ be a guarded RPS, and let $a : HA \to A$ be an H-CIA. Then the interpreted solution $e^\ddagger : VA \to A$ extends the CIA structure on A; more precisely, the $(H + V)$-algebra $[a, e^\ddagger] : (H + V)A \to A$ is a CIA for $H + V$.*

Remark 5.2. For the proof we need to recall some technical details. Recall that any guarded RPS $e : V \to T^{H+V}$ as in Definition 2.52 induces a natural

transformation

$$\bar{e} : T^{H+V} \to HT^{H+V} + \mathrm{Id}$$

(see [MM06], Lemma 6.9). The component $\bar{e}_A$ of this natural transformation at A is a flat equation morphism with parameters in A. Its unique solution in the H-CIA (A, a) is the Eilenberg-Moore algebra structure $\beta : T^{H+V}A \to A$ satisfying $[a, e^\ddagger] = \beta \cdot \kappa_A^{H+V}$ (this follows from [MM06], see Lemma 7.4 and the proof of Theorem 7.3).

Proof of Theorem 5.1. Let $m : X \to (H + V)X + A$ be a flat equation morphism. We need to prove that m has a unique solution s. As shortcut notations we shall write T for T^{H+V}, $\tau_X : (H + V)TX \to TX$ for the corresponding structure of a free CIA for $H+V$ as well as η and μ for the unit and multiplication of the monad T and $\kappa = \tau \cdot (H + V)\eta$ (cf. Theorem 2.47).

(1) Existence of a solution. Since TA is the final coalgebra for $(H + V)(-) + A$ by Corollary 2.35, we have a unique homomorphism $h : X \to TA$. We show that

$$s = (X \xrightarrow{h} TA \xrightarrow{\beta} A)$$

is a solution of m in the $(H + V)$-algebra $(A, [a, e^\ddagger])$. To see this, consider the following diagram:

$$\begin{array}{ccccc} X & \xrightarrow{\quad h \quad} & TA & \xrightarrow{\quad \beta \quad} & A \\ {\scriptstyle m}\downarrow & & \uparrow{\scriptstyle [\tau_A, \eta_A]} & & \uparrow{\scriptstyle [a, e^\ddagger, A]} \\ (H+V)X + A & \xrightarrow[(H+V)h+A]{} & (H+V)TA + A & \xrightarrow[(H+V)\beta+A]{} & (H+V)A + A \end{array}$$

The left-hand square commutes since h is an $((H + V)(-) + A)$-coalgebra homomorphism, and for the right-hand component of the right-hand square use the unit law $\beta \cdot \eta_A = \mathrm{id}_A$ of the Eilenberg-Moore algebra $\beta : TA \to A$. It remains to prove the commutativity of the left-hand component. This is established by inspecting the diagram below:

$$\begin{array}{ccccc} & & \overset{\tau_A}{\frown} & & \\ (H+V)TA & \xrightarrow[\kappa_{TA}]{} & TTA & \xrightarrow[\mu_A]{} & TA \\ {\scriptstyle (H+V)\beta}\downarrow & & {\scriptstyle T\beta}\downarrow & & \downarrow{\scriptstyle \beta} \\ (H+V)A & \xrightarrow[\kappa_A]{} & TA & \xrightarrow[\beta]{} & A \\ & & \underset{[a,e^\ddagger]}{\smile} & & \end{array} \tag{5.1}$$

Lemma 2.49 shows the commutativity of the upper part, for the lower one see Remark 5.2, the left-hand inner square commutes due to naturality of κ, and

the right-hand inner square by the multiplication law for the Eilenberg-Moore algebra $\beta : TA \to A$.

(2) Uniqueness of solutions. Since e is a guarded RPS, it factors through some $e' : V \to HT$ (cf. Definition 2.52). From e' and m we form a flat equation morphism $g : X + TX \to H(X + TX) + A$ w. r. t. H as follows. The left-hand component of g is

$$g \cdot \mathsf{inl} = (\, X \xrightarrow{m} HX + VX + A \xrightarrow{HX + e'_X + A} HX + HTX + A \xrightarrow{\mathsf{can}+A} H(X + TX) + A\,),$$

and the right-hand one is

$$g \cdot \mathsf{inr} = (TX \xrightarrow{\bar{e}_X} HTX + X \xrightarrow{[\mathsf{inl}\cdot H\mathsf{inr}, g\cdot\mathsf{inl}]} H(X + TX) + A)\,.$$

Since (A, a) is a CIA for H there exists a unique solution $g^\dagger : X + TX \to A$. Now let $s : X \to A$ be any solution of the flat equation morphism m in the $(H + V)$-algebra $[a, e^\ddagger] : (H + V)A \to A$. We will show below that $[s, \beta \cdot Ts] : X + TX \to A$ is a solution of g in the H-algebra (A, a). So since (A, a) is a CIA we have the following equation:

$$g^\dagger = [s, \beta \cdot Ts] : X + TX \to A\,. \tag{5.2}$$

Then s is uniquely determined by $g^\dagger$.

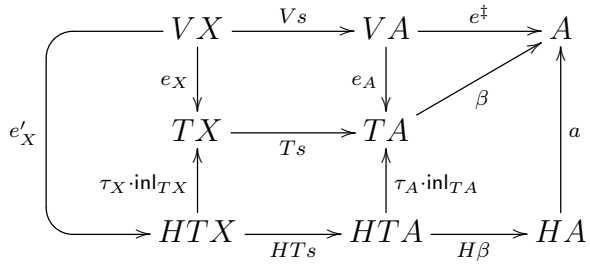
In order to prove equation (5.2) we need to verify that the following square commutes:

$$\begin{array}{ccc} X + TX & \xrightarrow{[s,\beta\cdot Ts]} & A \\ {\scriptstyle g}\downarrow & & \uparrow{\scriptstyle [a,A]} \\ H(X + TX) + A & \xrightarrow[H[s,\beta\cdot Ts]+A]{} & HA + A \end{array} \tag{5.3}$$

We shall verify the commutativity of the two coproduct components separately. For the left-hand component we consider the diagram below:

$$\begin{array}{ccccc} X & & \xrightarrow{\quad s \quad} & & A \\ {\scriptstyle m}\downarrow & & & & \uparrow{\scriptstyle [a,e^\ddagger,A]} \\ HX + VX + A & & \xrightarrow[Hs+Vs+A]{} & & HA + VA + A \\ {\scriptstyle HX+e'_X+A}\downarrow & & & & \\ HX + HTX + A & & & {\scriptstyle HA+[e^\ddagger,A]}\downarrow & \\ {\scriptstyle \mathsf{can}+A}\downarrow & \searrow^{[Hs,H(\beta\cdot Ts)]+A} & & & \\ H(X + TX) + A & & \xrightarrow[H[s,\beta\cdot Ts]+A]{} & & HA + A \end{array} \qquad \begin{array}{l} {\scriptstyle g\cdot\mathsf{inl}} \\ \\ \\ \\ \\ \\ \\ \end{array} \qquad \begin{array}{l} \\ \\ \\ \\ \\ \\ {\scriptstyle [a,A]} \\ \\ \end{array} \tag{5.4}$$

The left-hand part commutes by the definition of g, the right-hand part commutes trivially, the upper square commutes since s is a solution of m and the lower triangle commutes trivially, again. It remains to verify that the middle part commutes. We check the commutativity of this part componentwise: the left-hand and right-hand components commute trivially. We do not claim that the middle component commutes. However, in order to prove that the overall outside of (5.4) commutes, we need only show that this middle component commutes when extended by $[a, A] : HA + A \to A$. To see this consider the next diagram:

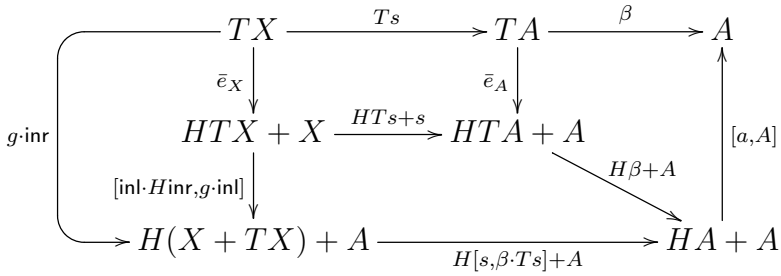


This diagram commutes: the left-hand part commutes since e is a guarded RPS; the upper and lower squares in the middle commute due to the naturality of e and of $\tau : (H + V)T \to T$ and $\mathsf{inl}T : HT \to (H + V)T$, respectively; the upper right-hand triangle commutes since $e^\ddagger$ is an interpreted solution of the RPS e. Finally, to see that the lower right-hand part commutes recall from diagram (5.1) that

$$[a, e^\ddagger] \cdot (H + V)\beta = \beta \cdot \tau_A \,.$$

Compose this last equation with the coproduct injection $\mathsf{inl}_{TA} : HTA \to (H + V)TA$ to obtain the desired commutativity.

Finally, we verify that the right-hand component of (5.3) commutes. Indeed, consider the diagram below:



The left-hand part commutes by the definition of g; the upper middle square commutes by the naturality of $\bar{e}$, the right-hand part of the above diagram

commutes since β is the solution of $\bar{e}_A$ in the CIA (A, a) (see Remark 5.2), and for the lower middle part we consider the components separately: the left-hand component clearly commutes by the functoriality of H, and for the right-hand component observe that it commutes when extended by $[a, A] : HA + A \to A$ (see diagram (5.4)). Thus, the outside of the above diagram commutes, and this completes the proof. □

Remark 5.3. The existence of a solution for m is clear: the Eilenberg-Moore algebra $\beta : TA \to A$ shows us that A is a complete Elgot algebra for $H + V$, and so A comes with a canonical choice of a solution for every flat equation morphism m, see [AMV06a, MM06]. However, since we need parts of the above existence proof for the uniqueness part anyway, we proved the existence part more concretely.

Remark 5.4. From $[a, e^\ddagger]$ being a CIA for $H + V$ it follows that $e^\ddagger$ is a CIA for V. Thus as a by-product of Theorem 5.1 we have proved interpreted solutions of RPS's in a CIA to be CIAs again. We remark that the copairing of two CIAs is not necessarily a CIA, consider e. g. the unary CIAs with carrier $\{1, \ldots, 10, \bot\}$ and the predecessor and successor operations, respectively.

Remark 5.5 (Compositionality of (ordinary) RPS's). Theorem 5.1 implies modularity of guarded RPS's as follows: operations obtained as solutions of guarded recursive program schemes can be used in subsequent guarded recursive operation definitions, which will still have unique solutions.
Moreover, this modularity property extends to a compositionality property of guarded RPS's as follows: let two guarded RPS's $e : V \to T^{H+V}$ and $g : W \to T^{(H+V)+W}$ with factors $e' : V \to HT^{H+V}$ and $g' : W \to (H + V)T^{(H+V)+W}$ be given. They can be composed into an RPS

$$h = \big(V + W \xrightarrow{[\mathsf{inr} * \eta^{H+V+W}, g']} (H + V)T^{H+V+W} \xrightarrow{[\kappa^{H+V+W} \cdot \mathsf{inl}, (\kappa^{H+V+W} \cdot \mathsf{inl})^\# \cdot e]T^{H+V+W}} T^{H+V+W}T^{H+V+W} \xrightarrow{\mu^{H+V+W}} T^{H+V+W} \big)$$

where $(\kappa^{H+V+W} \cdot \mathsf{inl})^\# : T^{H+V} \to T^{H+V+W}$ is the unique idealized monad morphism between CIMs extending $\kappa^{H+V+W} \cdot \mathsf{inl} : H + V \to T^{H+V+W}$, see Definition 2.46. Moreover, the composed RPS h is guarded as proved in Lemma 5.6 below. This is slightly more general than the composition of guarded RPS's in [MM09], where g needs to be guarded by H instead of $H + V$. Finally, the compositionality property of guarded RPS's is also proved in Lemma 5.6 below: one can either solve h directly or first take the solution $e^\ddagger$ of e and then solve g using $e^\ddagger$—both ways lead to the same

solution in the sense that $h^\ddagger = [e^\ddagger, g^\ddagger]$.
For the special case of interpreted RPS solutions in CIAs this strengthens the results in [MM09] (the interpreted Bekič-Scott law).

Lemma 5.6. *Let $e : V \to T^{H+V}$ and $g : W \to T^{H+V+W}$ be guarded RPS's, and let $h : V + W \to T^{H+V+W}$ be the RPS composed from e and g as in Remark 5.5. Then h is guarded, and its interpreted solution $h^\ddagger$ in a CIA (A, a) is given by $h^\ddagger = [e^\ddagger, g^\ddagger]$, where $e^\ddagger$ is the interpreted solutions of e in (A, a) and $g^\ddagger$ is the interpreted solution of g in $(A, [a, e^\ddagger])$.*

Proof. In order to simplify notation, let us denote the monad morphism $(\kappa^{H+V+W} \cdot \mathsf{inl})^\# : T^{H+V} \to T^{H+V+W}$ by γ.
The composed RPS h is guarded since it factors through

$$\begin{array}{rcl} h' = (\, V + W & \xrightarrow{[\mathsf{inr} * \eta^{H+V+W}, g']} & (H+V)T^{H+V+W} \\ & & \downarrow{\scriptstyle [H\eta^{H+V+W}, H\gamma \cdot e']T^{H+V+W}} \\ & & HT^{H+V+W}T^{H+V+W} \xrightarrow{H\mu^{H+V+W}} HT^{H+V+W}\,) \,. \end{array}$$

Indeed, we calculate

$$\begin{aligned} & \mu^{H+V+W} \cdot (\kappa^{H+V+W} \cdot \mathsf{inl})T^{H+V+W} \\ = {} & \tau^{H+V+W} \cdot \mathsf{inl}T^{H+V+W} \\ = {} & \tau^{H+V+W} \cdot \mathsf{inl}T^{H+V+W} \cdot H\mu^{H+V+W} \cdot H\eta^{H+V+W}T^{H+V+W} \end{aligned}$$

using Lemma 2.49 and one of the unit laws for the monad T^{H+V+W}, and

$$\begin{aligned}
&\mu^{H+V+W} \cdot (\gamma \cdot e)T^{H+V+W} \\
&= \mu^{H+V+W} \cdot (\gamma \cdot \tau^{H+V} \cdot \mathsf{inl}T^{H+V} \cdot e')T^{H+V+W} \\
&= \mu^{H+V+W} \cdot (\gamma \cdot \mu^{H+V} \cdot \kappa^{H+V}T^{H+V} \cdot \mathsf{inl}T^{H+V} \cdot e')T^{H+V+W} \\
&= \mu^{H+V+W} \cdot (\mu^{H+V+W} \cdot T^{H+V+W}\gamma \cdot \gamma T^{H+V} \\
&\quad \cdot \kappa^{H+V}T^{H+V} \cdot \mathsf{inl}T^{H+V} \cdot e')T^{H+V+W} \\
&= \mu^{H+V+W} \cdot (\mu^{H+V+W} \cdot T^{H+V+W}\gamma \cdot \kappa^{H+V+W}T^{H+V} \\
&\quad \cdot \mathsf{inl}T^{H+V} \cdot \mathsf{inl}T^{H+V} \cdot e')T^{H+V+W} \\
&= \mu^{H+V+W} \cdot (\tau^{H+V+W} \cdot (H+V+W)\gamma \cdot \mathsf{inl}T^{H+V} \cdot e')T^{H+V+W} \\
&= \mu^{H+V+W} \cdot \tau^{H+V+W}T^{H+V+W} \cdot (H+V+W)\gamma T^{H+V+W} \\
&\quad \cdot \mathsf{inl}T^{H+V}T^{H+V+W} \cdot e'T^{H+V+W} \\
&= \tau^{H+V+W} \cdot (H+V+W)\mu^{H+V+W} \cdot (H+V+W)\gamma T^{H+V+W} \\
&\quad \cdot \mathsf{inl}T^{H+V}T^{H+V+W} \cdot e'T^{H+V+W} \\
&= \tau^{H+V+W} \cdot \mathsf{inl}T^{H+V+W} \cdot H\mu^{H+V+W} \cdot H\gamma T^{H+V+W} \cdot e'T^{H+V+W} \\
&= \tau^{H+V+W} \cdot \mathsf{inl}T^{H+V+W} \cdot H\mu^{H+V+W} \cdot (H\gamma \cdot e')T^{H+V+W}
\end{aligned}$$

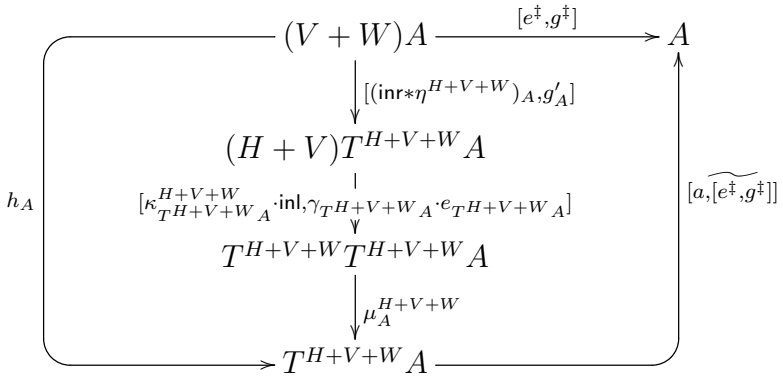
using the guardedness of e, Lemma 2.49, that γ is the unique idealized monad morphism extending $\kappa^{H+V+W} \cdot \mathsf{inl}$, Remark 2.48 and naturality of $\mathsf{inl} : H \to H + V + W$. From these two equations and the definitions of h and h' we infer

$$\begin{aligned}
h &= \mu^{H+V+W} \cdot [\kappa^{H+V+W} \cdot \mathsf{inl}, \gamma \cdot e]T^{H+V+W} \cdot [\mathsf{inr} * \eta^{H+V+W}, g'] \\
&= \tau^{H+V+W} \cdot \mathsf{inl}T^{H+V+W} \cdot H\mu^{H+V+W} \\
&\quad \cdot [H\eta^{H+V+W}, H\gamma \cdot e']T^{H+V+W} \cdot [\mathsf{inr} * \eta^{H+V+W}, g'] \\
&= \tau^{H+V+W} \cdot \mathsf{inl}T^{H+V+W} \cdot h'
\end{aligned}$$

as desired.

Now let (A, a) be a CIA for H, let $e^\ddagger$ be the interpreted solution of e in (A, a) and let $g^\ddagger$ be the interpreted solution of g in $(A, [a, e^\ddagger])$ (recall from

Theorem 5.1 that $[a, e^\ddagger]$ is a CIA for $H+V$). Consider the following diagram:



The left-hand part is the definition of h. The right-hand part commutes since both coproduct components commute: the left-hand component commutes since we have

$$\begin{aligned}
&[a, \widetilde{[e^\ddagger, g^\ddagger]}] \cdot \mu^{H+V+W}_A \cdot [\kappa^{H+V+W}_{T^{H+V+W}A} \cdot \mathsf{inl}, \gamma_{T^{H+V+W}A} \cdot e_{T^{H+V+W}A}] \\
&\cdot (\mathsf{inr} * \eta^{H+V+W})_A \\
&= [a, \widetilde{[e^\ddagger, g^\ddagger]}] \cdot \mu^{H+V+W}_A \cdot \gamma_{T^{H+V+W}A} \cdot e_{T^{H+V+W}A} \cdot V\eta^{H+V+W}_A \\
&= [a, \widetilde{[e^\ddagger, g^\ddagger]}] \cdot \gamma_A \cdot e_A \\
&= \widetilde{[a, e^\ddagger]} \cdot e_A \\
&= e^\ddagger
\end{aligned}$$

where we use naturality of e and γ and one of the unit laws for the monad T^{H+V+W}, that $[a, \widetilde{[e^\ddagger, g^\ddagger]}] \cdot \gamma_A = \widetilde{[a, e^\ddagger]}$ (which holds since both sides are the unique homomorphism between the free CIA $\tau^{H+V}_A : (H+V)T^{H+V}A \to T^{H+V}A$ and the CIA $[a, e^\ddagger] : (H+V)A \to A$ extending id_A, cf. the proof of Lemma 4.17), and that $e^\ddagger$ is a solution of e in (A, a). The right-hand

component commutes since

$$\begin{aligned}
&\widetilde{[a,[e^\ddagger,g^\ddagger]]} \cdot \mu_A^{H+V+W} \cdot [\kappa^{H+V+W}_{T^{H+V+W}A} \cdot \mathsf{inl}, \gamma_{T^{H+V+W}A} \cdot e_{T^{H+V+W}A}] \cdot g'_A \\
\overset{(*)}{=}\ & [a,e^\ddagger] \cdot (H+V)\widetilde{[[a,e^\ddagger],g^\ddagger]} \cdot g'_A \\
=\ & [[a,e^\ddagger],g^\ddagger] \cdot (H+V+W)\widetilde{[[a,e^\ddagger],g^\ddagger]} \cdot \mathsf{inl} \cdot g'_A \\
=\ & \widetilde{[[a,e^\ddagger],g^\ddagger]} \cdot T^{H+V+W}\widetilde{[[a,e^\ddagger],g^\ddagger]} \cdot \kappa^{H+V+W}_{T^{H+V+W}A} \cdot \mathsf{inl} \cdot g'_A \\
=\ & \widetilde{[[a,e^\ddagger],g^\ddagger]} \cdot \mu_A^{H+V+W} \cdot \kappa^{H+V+W}_{T^{H+V+W}A} \cdot \mathsf{inl} \cdot g'_A \\
=\ & \widetilde{[[a,e^\ddagger],g^\ddagger]} \cdot \tau_A^{H+V+W} \cdot \mathsf{inl} \cdot g'_A \\
=\ & \widetilde{[[a,e^\ddagger],g^\ddagger]} \cdot g_A \\
=\ & g^\ddagger
\end{aligned}$$

where we use $\widetilde{[a,[e^\ddagger,g^\ddagger]]} \cdot \kappa_A^{H+V+W} = [a,[e^\ddagger,g^\ddagger]]$, that $\widetilde{[[a,e^\ddagger],g^\ddagger]}$ is an Eilenberg-Moore algebra, Lemma 2.49, guardedness of g and that $g^\ddagger$ is a solution of g in $(A,[a,e^\ddagger])$. The equation marked by $(*)$ holds since we can remove g'_A and have

$$\begin{aligned}
&\widetilde{[a,[e^\ddagger,g^\ddagger]]} \cdot \mu_A^{H+V+W} \cdot \kappa^{H+V+W}_{T^{H+V+W}A} \cdot \mathsf{inl} \\
=\ & \widetilde{[a,[e^\ddagger,g^\ddagger]]} \cdot T^{H+V+W}\widetilde{[a,[e^\ddagger,g^\ddagger]]} \cdot \kappa^{H+V+W}_{T^{H+V+W}A} \cdot \mathsf{inl} \\
=\ & [a,e^\ddagger,g^\ddagger] \cdot (H+V+W)\widetilde{[a,[e^\ddagger,g^\ddagger]]} \cdot \mathsf{inl} \\
=\ & a \cdot H\widetilde{[[a,e^\ddagger],g^\ddagger]}
\end{aligned}$$

for the left-hand coproduct component, where we use the properties of the Eilenberg-Moore algebra $\widetilde{[a,[e^\ddagger,g^\ddagger]]}$, and

$$\begin{aligned}
&\widetilde{[a,[e^\ddagger,g^\ddagger]]} \cdot \mu_A^{H+V+W} \cdot \gamma_{T^{H+V+W}A} \cdot e_{T^{H+V+W}A} \\
=\ & \widetilde{[a,[e^\ddagger,g^\ddagger]]} \cdot T^{H+V+W}\widetilde{[a,[e^\ddagger,g^\ddagger]]} \cdot \gamma_{T^{H+V+W}A} \cdot e_{T^{H+V+W}A} \\
=\ & \widetilde{[a,e^\ddagger]} \cdot T^{H+V}\widetilde{[a,[e^\ddagger,g^\ddagger]]} \cdot e_{T^{H+V+W}A} \\
=\ & e^\ddagger \cdot V\widetilde{[[a,e^\ddagger],g^\ddagger]}
\end{aligned}$$

for the right-hand one. Here we use that $\widetilde{[a,[e^\ddagger,g^\ddagger]]}$ is an Eilenberg-Moore algebra, naturality of γ and that we have $\widetilde{[a,[e^\ddagger,g^\ddagger]]} \cdot \gamma_A = \widetilde{[a,e^\ddagger]}$ (which holds since both sides are the unique homomorphism between the free CIA $\tau_A^{H+V} : (H+V)T^{H+V}A \to T^{H+V}A$ and the CIA $[a,e^\ddagger] : (H+V)A \to A$

extending id_A, cf. the proof of Lemma 4.17), and naturality of e and that $e^\ddagger$ is a solution of e in (A, a).
Thus we proved that the above diagram commutes, i. e. that $[e^\ddagger, g^\ddagger]$ is a solution of h in (A, a). The uniqueness of such solutions yields $h^\ddagger = [e^\ddagger, g^\ddagger]$. □

Coming back to our setting in Chapter 4, let $\ell : K(H \times \mathrm{Id}) \to HF$ be an abstract GSOS rule, where F is the free monad over K (or, more generally, let λ be a distributive law of an arbitrary monad M over the cofree copointed functor $H \times \mathrm{Id}$, see Section 4.4). Assume furthermore that the composites HM and MHM are iteratable. By applying Theorems 2.54 and 5.1 to the CIAs $k : HMC \to C$ from Theorem 4.7 and $k' : MHMC \to C$ from Theorem 4.8 respectively, we get two more solution theorems for free:

Corollary 5.7. *Every guarded RPS $e : V \to T^{HM+V}$ has a unique interpreted solution in the CIA (C, k), and this solution extends the CIA structure on C.*

Corollary 5.8. *Every guarded RPS $e : V \to T^{MHM+V}$ has a unique interpreted solution in the CIA (C, k'), and this solution extends the CIA structure on C.*

Remark 5.9. The format of a guarded RPS $V \to T^{HM+V}$, for which unique solutions in the CIA (C, k) exist by Corollary 5.7, comprises (for any $\ell : K(H \times \mathrm{Id}) \to HF$ or even $\lambda : M(H \times \mathrm{Id}) \to (H \times \mathrm{Id})M$) the format of a guarded RPS $V \to T^{H+V}$, for which unique solutions in the CIA (C, c^{-1}) exist by Theorem 2.54: every guarded RPS $e : V \to T^{H+V}$ can be considered as an RPS $(\kappa^{HM+V} \cdot (H\eta^M + V))^\# \cdot e : V \to T^{HM+V}$, where $(\kappa^{HM+V} \cdot (H\eta^M + V))^\# : T^{H+V} \to T^{HM+V}$ is the unique idealized monad morphism between CIMs extending $\kappa^{HM+V} \cdot (H\eta^M + V) : H + V \to T^{HM+V}$, see Definition 2.46. It is straightforward to check that this RPS is guarded, too, and that the respective solutions coincide (i. e., the solution is preserved by the construction). Moreover, the format of a guarded RPS $V \to T^{MHM+V}$, for which unique solutions in the CIA (C, k') exist by Corollary 5.8, comprises the format of a guarded RPS $V \to T^{HM+V}$: every guarded RPS $e : V \to T^{HM+V}$ can be considered as an RPS $(\kappa^{MHM+V} \cdot (\eta^M HM + V))^\# \cdot e : V \to T^{MHM+V}$, where $(\kappa^{MHM+V} \cdot (\eta^M HM + V))^\# : T^{HM+V} \to T^{MHM+V}$ is the unique idealized monad morphism between CIMs extending $\kappa^{MHM+V} \cdot (\eta^M HM + V) : HM + V \to T^{MHM+V}$. Again one checks that this RPS is guarded and that the construction preserves the solution.

5.1.2 ℓ-RPS's and Sandwiched ℓ-RPS's

Even with all the results we have seen so far, we are still not able to obtain operations such as the shuffle product $\otimes$ on streams as a unique solution since its definition refers to the behavior of the arguments of the function: it is given by the recursive specification

$$r.x \otimes s.y = (r \cdot s).((x \otimes s.y) + (r.x \otimes y)) \,. \tag{5.5}$$

Notice also that this specification makes use of the componentwise stream addition $+$, so this operation is assumed as given or as previously specified, and the specification of $\otimes$ is built on top of the specification of $+$.

In this section we start with an abstract GSOS rule $\ell : K(H \times \mathrm{Id}) \to HF^K$ specifying all given operations (such as the stream addition). We introduce a special form of rule called *recursive program scheme w. r. t.* ℓ (or, *ℓ-RPS*, for short) specifying new operations in terms of given ones (such as the shuffle product of streams). We prove that every ℓ-RPS has a unique solution in the final H-coalgebra C, and this solution extends the CIA structure for HF^K on C given by Theorem 4.7—this is a result similar to the one given in Theorem 5.1 for ordinary RPS's.

We show that every ℓ-RPS easily gives rise to a "composed" abstract GSOS rule, and so the results in this section are essentially an application of the work in [Bar03, Bar04] and our results in Section 4.1.2. Based on this, we prove a modularity and compositionality principle for ℓ-RPS's.

Finally, we introduce an even more general format called *sandwiched ℓ-RPS* and prove similar results as for ℓ-RPS's.

Assumption 5.10. In addition to Assumption 4.1 we assume that an abstract GSOS rule $\ell : K(H \times \mathrm{Id}) \to HF^K$ is given. We shall also use free monads of other functors than K, and we follow the convention that whenever we write F^G for a functor G we assume that the free monad F^G exists and is given objectwise by free algebras for G, cf. Theorem 2.22.

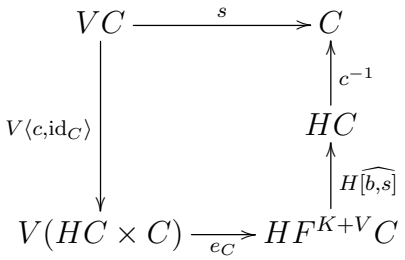
ℓ-RPS's

Definition 5.11. A *recursive program scheme w. r. t.* ℓ (shortly, *ℓ-RPS*) is a natural transformation

$$e : V(H \times \mathrm{Id}) \to HF^{K+V} \,.$$

An *interpreted solution* of e in C is a V-algebra $s : VC \to C$ such that

the diagram



(5.6)

commutes where $b : KC \to C$ is the ℓ-interpretation in C, cf. Theorem 3.17.

Remark 5.12. In Definition 5.11 we define ℓ-RPS's to have finite terms on the right-hand side of equtations, i. e. we use the free monad F^{K+V} instead of T^{K+V} as for ordinary RPS's (see Definition 2.52). The reason is that using finite terms allows us to prove solutions of ℓ-RPS's to exist uniquely in Theorem 5.16 below. Infinite terms may lead to ℓ-RPS's where no solution exists: consider e. g. $V = (-)^2$, $H = \mathbb{R} \times -$ and K arbitrary and the ℓ-RPS $V(H \times \mathrm{Id}) \to HF^{K+V}$ given componentwise for each set X by the equation

$$f((r, x', x), (s, y', y)) = (r + s, t)$$

where $r, s \in \mathbb{R}$, $x, x', y, y' \in X$ and where t is the infinite term $f(f(f(f(f(\dots), y'), x'), y'), x')$. Then $f([1, 1, \dots], [1, 1, \dots])$ is not defined since all of its elements except the head element would have to equal ∞.

Let us justify the name "ℓ-RPS" by relating this concept to ordinary RPS's. Since we use finite terms, i. e. free monads, for our ℓ-RPS's, we compare them to the finite term RPS's $V \to F^{H+V}$ which are a subset of the infinite term RPS's $V \to T^{H+V}$ from Definition 2.52 via the unique monad morphism $F^{H+V} \to T^{H+V}$ extending $\kappa^{H+V} : H + V \to T^{H+V}$. And clearly, since solutions of ℓ-RPS's are only defined on C, we compare to interpreted solutions of ordinary RPS's in a CIA on C. With these a priori restrictions, the following lemma shows that ℓ-RPS's generalize ordinary RPS's.

Lemma 5.13. *Every guarded RPS $e : V \to F^{H+V}$ with the unique interpreted solution $e^\ddagger : VC \to C$ in the CIA $c^{-1} : HC \to C$ gives rise to an ℓ-RPS that also has the solution $e^\ddagger$.*

Proof. Let ℓ be the abstract GSOS rule $H\kappa^H \cdot H\pi_0 : H(H \times \mathrm{Id}) \to HF^H$ (where $K = H$). Its ℓ-interpretation is easily seen to be $c^{-1} : HC \to C$ by checking the commutativity of diagram (3.1).

Now let $e : V \to F^{H+V}$ be a guarded RPS, i. e. we have

$$e = (\ V \xrightarrow{e'} HF^{H+V} \xrightarrow{\mathsf{in}lF^{H+V}} (H + V)F^{H+V} \xrightarrow{\phi^{H+V}} F^{H+V}\ )$$

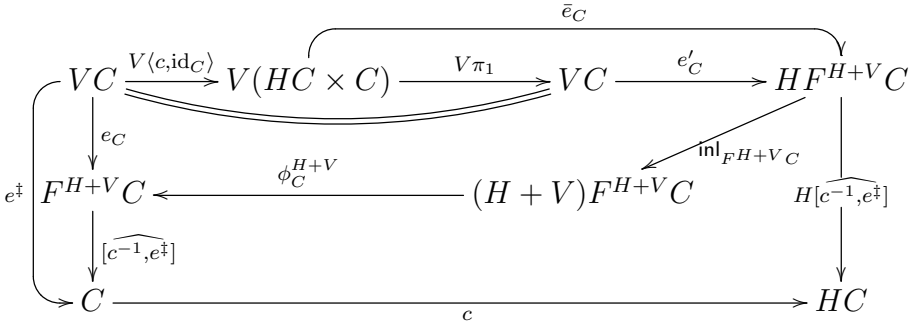
for some e'. Then we form the following ℓ-RPS $\bar{e}$ for the above abstract GSOS rule ℓ:

$$\bar{e} = (\ V(H \times \mathrm{Id}) \xrightarrow{V\pi_1} V \xrightarrow{e'} HF^{H+V}\ )$$

Assume that $e^\ddagger : VC \to C$ is the interpreted solution of e in the CIA $c^{-1} : HC \to C$, i. e. the triangle

$$\begin{array}{ccc} VC & \xrightarrow{e^\ddagger} & C \\ {\scriptstyle e_C}\downarrow & \nearrow_{\widehat{[c^{-1},e^\ddagger]}} & \\ F^{H+V}C & & \end{array} \tag{5.7}$$

commutes. Then we prove that $e^\ddagger$ also is a solution of the ℓ-RPS $\bar{e}$ by showing that the outside of the diagram



commutes. Indeed, the upper part is the definition of $\bar{e}$, the left-hand part is (5.7), the big upper part is the guardedness of e, the upper left-hand part commutes due to $\pi_1 \cdot \langle c, \mathrm{id}_C\rangle = \mathrm{id}_C$, and we verify the commutativity of the lower part as follows:

$$\begin{aligned} & c \cdot \widehat{[c^{-1}, e^\ddagger]} \cdot \phi_C^{H+V} \cdot \mathsf{inl}_{F^{H+V}C} \\ = {} & c \cdot \widehat{[c^{-1}, e^\ddagger]} \cdot \mu_C^{H+V} \cdot \kappa^{H+V}_{F^{H+V}C} \cdot \mathsf{inl}_{F^{H+V}C} \\ = {} & c \cdot \widehat{[c^{-1}, e^\ddagger]} \cdot F^{H+V}\widehat{[c^{-1}, e^\ddagger]} \cdot \kappa^{H+V}_{F^{H+V}C} \cdot \mathsf{inl}_{F^{H+V}C} \\ = {} & c \cdot [c^{-1}, e^\ddagger] \cdot (H+V)\widehat{[c^{-1}, e^\ddagger]} \cdot \mathsf{inl}_{F^{H+V}C} \\ = {} & c \cdot c^{-1} \cdot H\widehat{[c^{-1}, e^\ddagger]} \\ = {} & H\widehat{[c^{-1}, e^\ddagger]} \end{aligned}$$

by Lemma 2.24, the properties of the Eilenberg-Moore algebra $\widehat{[c^{-1}, e^\ddagger]}$ and naturality of κ^{H+V}. Finally all inner parts commute, thus the outside commutes as desired. □

Next we also relate ℓ-RPS's to ℓ-equations (see Definition 4.2): it turns out that the latter are a special case of ℓ-RPS's.

Lemma 5.14. *Every ℓ-equation is an ℓ-RPS.*

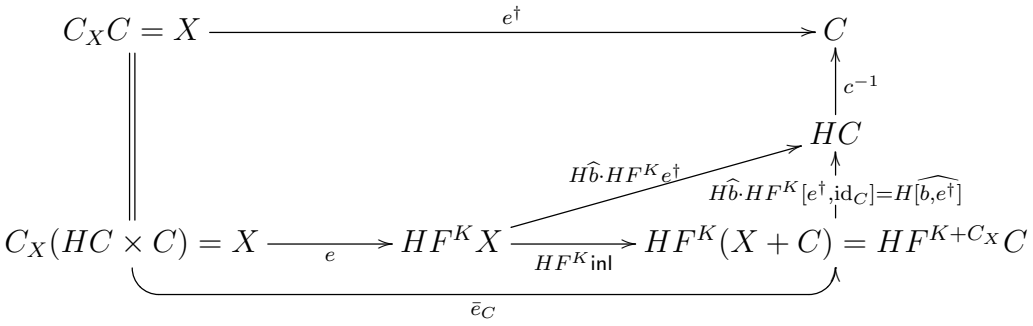
Proof. We prove that every ℓ-equation $e : X \to HF^K X$ gives rise to an ℓ-RPS $\bar{e} : V(H \times \mathrm{Id}) \to HF^{K+V}$ where $V = C_X$ is the constant functor on X (this means that all operations newly defined by $\bar{e}$ are constants), and that the respective solutions coincide.

First, we show that any ℓ-equation $e : X \to HF^K X$ gives rise to a natural transformations $\bar{e} : C_X \to HF^K(C_X + \mathrm{Id})$; since the functors $F^K(C_X + \mathrm{Id})$ and F^{K+C_X} are naturally isomorphic as proved in Lemma 5.15 below and since $C_X = C_X(H \times \mathrm{Id})$, $\bar{e}$ also is an ℓ-RPS $\bar{e} : C_X(H \times \mathrm{Id}) \to HF^{K+C_X}$. We define the Y-component of $\bar{e}$ by

$$\bar{e}_Y = (\ C_X Y = X \xrightarrow{e} HF^K X \xrightarrow{HF^K \mathsf{inl}} HF^K(X+Y) = HF^K(C_X + \mathrm{Id})Y\ ).$$

Then clearly $\bar{e}$ is a natural transformation.

Second, we prove that the solution $e^\dagger : X \to C$ of e is an interpreted solution of $\bar{e}$. Clearly, morphisms $e^\dagger : X \to C$ correspond to (or actually are) morphisms $C_X C \to C$. Consider the following diagram:

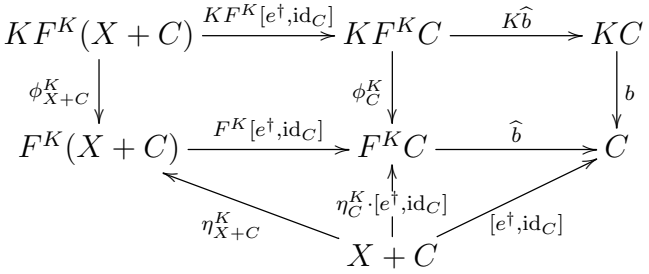


The lower part commutes by the definition of $\bar{e}$, the big upper left-hand part commutes since $e^\dagger$ is a solution of the ℓ-equation e, and the remaining triangle commutes by $[e^\dagger, \mathrm{id}_C] \cdot \mathsf{inl} = e^\dagger$. Finally, to prove the equation $H\widehat{b} \cdot HF^K[e^\dagger, \mathrm{id}_C] = H\widehat{[b, e^\dagger]}$ it suffices to show

$$\widehat{b} \cdot F^K[e^\dagger, \mathrm{id}_C] = \widehat{[b, e^\dagger]}\,. \tag{5.8}$$

This equation follows from the proof of Lemma 5.15 below as we show next:

observe from the diagram



that the left-hand side of (5.8) is the unique homomorphism between the free K-algebra on $X+C$ and the K-algebra (C,b) such that $\widehat{b}\cdot F^K[e^\dagger,\mathrm{id}_C]\cdot\eta^K_{X+C} = [e^\dagger,\mathrm{id}_C]$. Indeed, the left-hand parts commute by the naturality of ϕ^K and η^K, and the right-hand parts commute by the properties of the Eilenberg-Moore algebra $\widehat{b}$ as explained below Notation 3.4. The right-hand side of (5.8) is the unique homomorphism between the free $(K+C_X)$-algebra on C and the $(K+C_X)$-algebra $(C,[b,e^\dagger])$ such that $\widehat{[b,e^\dagger]}\cdot\eta^{K+C_X}_C = \mathrm{id}_C$. This follows from the properties of the Eilenberg-Moore algebra $\widehat{[b,e^\dagger]}$, see again below Notation 3.4. Setting $Y=C$, $(A,a)=(C,[b,e^\dagger])$ and $u=\mathrm{id}_C$ in the part of the proof of Lemma 5.15 showing that $F^K(X+Y)$ is the free F^{K+C_X}-algebra on Y, we conclude that both sides are the unique homomorphism h from that proof. □

Lemma 5.15. *The functors $F^K(C_X+\mathrm{Id})$ and F^{K+C_X} are naturally isomorphic.*

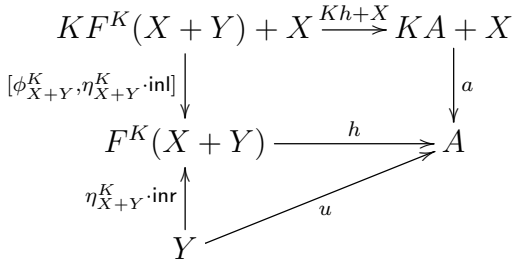
Proof. Recall from Assumption 3.1 that for every object Y the free K-algebra on Y exists. It follows that for every object Y also the free $(K+C_X)$-algebra on Y exists: in fact, it is given by

$$(F^{K+C_X}Y,\phi^{K+C_X}_Y) = (F^K(X+Y),[\phi^K_{X+Y},\eta^K_{X+Y}\cdot\mathsf{inl}])$$

together with the universal morphism $\eta^{K+C_X}_Y = \eta^K_{X+Y}\cdot\mathsf{inr}$.

In order to prove this, fix some Y, let (A,a) be any $(K+C_X)$-algebra and let $u: Y\to A$ be any morphism. The following diagram shows that the unique homomorphism $h: F^K(X+Y)\to A$ between the free K-algebra $(F^K(X+Y),\phi^K_{X+Y})$ on $X+Y$ and the K-algebra $(A,a\cdot\mathsf{inl})$ such that $h\cdot\eta^K_{X+Y} = [a\cdot\mathsf{inr},u]: X+Y\to A$ is at the same time the unique homomorphism between the $(K+C_X)$-algebra $(F^K(X+Y),[\phi^K_{X+Y},\eta^K_{X+Y}\cdot\mathsf{inl}])$ and the $(K+C_X)$-

algebra (A, a) such that $h \cdot \eta^K_{X+Y} \cdot \mathsf{inr} = u : Y \to A$:



(5.9)

The left-hand coproduct component of the upper square commutes since h is a homomorphism of K-algebras, and the right-hand one commutes since $a \cdot \mathsf{inr} = [a \cdot \mathsf{inr}, u] \cdot \mathsf{inl} = h \cdot \eta^K_{X+Y} \cdot \mathsf{inl}$. The lower triangle commutes since $u = [a \cdot \mathsf{inr}, u] \cdot \mathsf{inr} = h \cdot \eta^K_{X+Y} \cdot \mathsf{inr}$.

Conversely, any homomorphism h' between the $(K + C_X)$-algebras $(F^K(X+Y), [\phi^K_{X+Y}, \eta^K_{X+Y} \cdot \mathsf{inl}])$ and (A, a) such that $h \cdot \eta^K_{X+Y} \cdot \mathsf{inr} = u$ is at the same time a homomorphism between the free K-algebra $(F^K(X+Y), \phi^K_{X+Y})$ on $X + Y$ and the K-algebra $(A, a \cdot \mathsf{inl})$ such that $h \cdot \eta^K_{X+Y} = [a \cdot \mathsf{inr}, u] : X + Y \to A$. Indeed, then the upper and lower parts of diagram (5.9) commute with h replaced by h'. Then the left-hand coproduct component of the upper part proves h' to be a homomorphism between the free K-algebra and the K-algebra $(A, a \cdot \mathsf{inl})$. Moreover, taking the right-hand component of the upper part together with the lower part, we obtain $h' \cdot \eta^K_{X+Y} = [a \cdot \mathsf{inr}, u]$. By the uniqueness of such K-algebra homomorphisms, we conclude $h' = h$. Thus h is the unique homomorphism between the the $(K + C_X)$-algebras $(F^K(X + Y), [\phi^K_{X+Y}, \eta^K_{X+Y} \cdot \mathsf{inl}])$ and (A, a) such that $h \cdot \eta^K_{X+Y} \cdot \mathsf{inr} = u$.

This proves that $(F^K(X + Y), [\phi^K_{X+Y}, \eta^K_{X+Y} \cdot \mathsf{inl}])$ is the free $(K + C_X)$-algebra on Y with universal arrow $\eta^K_{X+Y} \cdot \mathsf{inr} : Y \to F^K(X + Y)$.

We conclude that the functors $F^K(C_X + \mathrm{Id})$ and F^{K+C_X} assign to every object Y the same object $F^K(X + Y) = F^{K+C_X}Y$. We still need to prove that they assign to every morphism $f : Y \to Z$ the same morphism, i. e. that $F^K(\mathrm{id}_X + f) = F^{K+C_X} f$. We do so by observing that the left-hand side is the unique homomorphism between the free K-algebras on $X + Y$ and $X + Z$ such that $F^K(\mathrm{id}_X + f) \cdot \eta^K_{X+Y} = \eta^K_{X+Z} \cdot (\mathrm{id}_X + f)$, and that the right-hand side is the unique homomorphism between the free $(K + C_X)$-algebras on Y and Z such that $F^{K+C_X} f \cdot \eta^{K+C_X}_Y = \eta^{K+C_X}_Z \cdot f$. Setting $(A, a) = (F^K(X + Z), [\phi^K_{X+Z}, \eta^K_{X+Z} \cdot \mathsf{inl}])$ and $u = \eta^K_{X+Z} \cdot \mathsf{inr} \cdot f$ in our above proof showing that $F^K(X+Y)$ is the free F^{K+C_X}-algebra on Y, we conclude that both sides are the unique homomorphism h from that proof. □

Now we state our main result about ℓ-RPS's.

Theorem 5.16. (Unique solutions of ℓ-RPS's extend the initial CIA)**.** *For every ℓ-RPS there exists a unique interpreted solution s in C. In addition, s extends the CIA structure on C, i. e., the following is a CIA for HF^{K+V}:*

$$HF^{K+V}C \xrightarrow{H\widehat{[b,s]}} HC \xrightarrow{c^{-1}} C\,. \tag{5.10}$$

Notice that this Theorem generalizes (due to Lemma 5.14) Theorem 4.3. Before we proceed to the proof we set up some notation, provide the construction of the unique solution s and establish a technical lemma.

Notations 5.17. 1. Let G and G' be endofunctors on our base category such that the free monads F^G, $F^{G'}$ and $F^{G+G'}$ exist. The coproduct injections $\mathsf{inl} : G \to G + G' \leftarrow G' : \mathsf{inr}$ lift to monad morphisms on the corresponding free monads, and we denote those monad morphisms by $\widehat{\mathsf{inl}} : F^G \to F^{G+G'} \leftarrow F^{G'} : \widehat{\mathsf{inr}}$.

2. Recall from Theorem 2.22 that for a functor K on our base category we write $\phi^K : KF^K \to F^K$ and $\eta^K : \mathrm{Id} \to F^K$ for (the natural transformations given by) the structures and universal morphisms of the free K-algebras as well as $\mu^K : F^K F^K \to F^K$ and $\kappa^K : K \to F^K$ for the multiplication and universal natural transformation of the free monad F^K on K.

Remarks 5.18. 1. Notice that the monad morphism $\widehat{\mathsf{inl}} : F^G \to F^{G+G'}$ is uniquely determined by the commutativity of the following square of natural transformations:

$$\begin{array}{ccc} G & \xrightarrow{\kappa^G} & F^G \\ {\scriptstyle \mathsf{inl}}\downarrow & & \downarrow{\scriptstyle \widehat{\mathsf{inl}}} \\ G + G' & \xrightarrow[\kappa^{G+G'}]{} & F^{G+G'} \end{array}$$

2. Recall from Lemma 3.3 and Notation 3.4 that for every G-algebra (A, a) we have the corresponding Eilenberg-Moore algebra $\widehat{a} : F^G A \to A$ and that $a = \widehat{a} \cdot \kappa^G_A$. Moreover, recall from Lemma 3.3 that the categories of G-algebras and of Eilenberg-Moore algebras for F^G are isomorphic; more precisely, $a \mapsto \widehat{a}$ and precomposition with κ^G_A extend to mutually inverse functors.

3. Combining parts (1) and (2) of this remark, we see that for every $(G + G')$-algebra $a : (G + G')A \to A$ the equation $\widehat{a} \cdot \widehat{\mathsf{inl}}_A = \widehat{a \cdot \mathsf{inl}_A}$ holds. Indeed, both sides are equal when precomposed with κ^G_A:

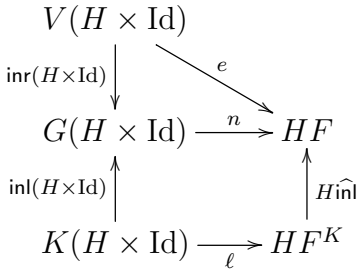
$$\widehat{a} \cdot \widehat{\mathsf{inl}}_A \cdot \kappa^G_A = \widehat{a} \cdot \kappa^{G+G'}_A \cdot \mathsf{inl}_A = a \cdot \mathsf{inl}_A = \widehat{a \cdot \mathsf{inl}_A} \cdot \kappa^G_A\,.$$

If we make the coproduct algebra structure explicit as in $a = [a_0, a_1] : FA + GA \to A$, we obtain

$$\widehat{[a_0, a_1]} \cdot \widehat{\mathsf{inl}}_A = \widehat{a_0} \qquad \text{and} \qquad \widehat{[a_0, a_1]} \cdot \widehat{\mathsf{inr}}_A = \widehat{a_1}\,.$$

Throughout the rest of this section we are going to write G for $K + V$ and F for F^{K+V}.

Construction 5.19. Let $e : V(H \times \mathrm{Id}) \to HF$ be an ℓ-RPS. This gives an abstract GSOS rule $n : G(H \times \mathrm{Id}) \to HF$ defined on its coproduct components as displayed below:



We write $a : GC \to C$ for the n-interpretation in C (see Theorem 3.17). Define

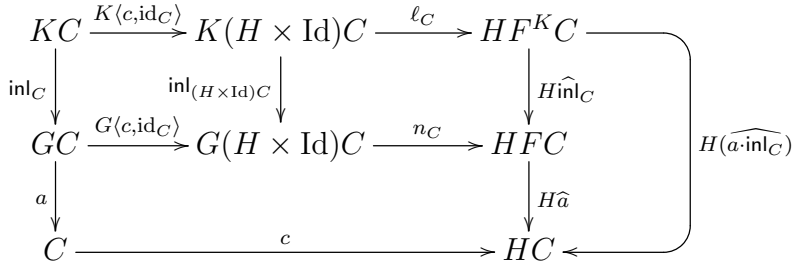
$$s = (VC \xrightarrow{\mathsf{inr}_C} GC \xrightarrow{a} C)\,.$$

We shall prove that s is the unique solution of e in C.

Lemma 5.20. *Let $b : KC \to C$ be the interpretation of $\ell : K(H \times \mathrm{Id}) \to HF^K$. Then for $a : GC \to C$ from Construction 5.19 we have*

$$b = (KC \xrightarrow{\mathsf{inl}_C} GC \xrightarrow{a} C)\,.$$

Proof. Consider the diagram below:



The lower square commutes since $a : GC \to C$ is the n-interpretation in C and the upper right-hand one by the definition of n (cf. Construction 5.19).

The upper left-hand square commutes by the naturality of $\mathsf{inl} : K \to G$, and for the right-hand part we remove H and use Remark 5.18(3). Thus the outside commutes.

Now recall that $b : KC \to C$ is uniquely determined by the commutativity of the diagram in Theorem 3.17. Thus, $a \cdot \mathsf{inl}_C = b$ holds as desired. □

Corollary 5.21. *The n-interpretation in C is $a = [b, s] : GC \to C$, thus*

$$\widehat{a} = \widehat{[b, s]} : FC \to C\,.$$

Proof of Theorem 5.16. We are ready to prove that s in Construction 5.19 is the unique interpreted solution of the ℓ-RPS $e : V(H \times \mathrm{Id}) \to HF$ in C.

(1) We prove that s is an interpreted solution of e in C. Indeed, consider the diagram

$$\begin{array}{ccccc}
VC & \xrightarrow{V\langle c,\mathrm{id}_C\rangle} & V(H\times\mathrm{Id})C & & \\
\downarrow{\scriptstyle \mathsf{inr}_C} & & \downarrow{\scriptstyle \mathsf{inr}_{(H\times\mathrm{Id})C}} & \searrow^{e_C} & \\
GC & \xrightarrow{G\langle c,\mathrm{id}_C\rangle} & G(H\times\mathrm{Id})C & \xrightarrow{n_C} & HFC \\
\downarrow{\scriptstyle a} & & & & \downarrow{\scriptstyle H\widehat{a}} \\
C & & \xrightarrow{\qquad\qquad c\qquad\qquad} & & HC\,.
\end{array}
\qquad s \qquad (5.11)$$

The lower square commutes since $a : GC \to C$ is the n-interpretation in C, and the upper right-hand triangle by the definition of n (cf. Construction 5.19). The upper left-hand square commutes by naturality of inr, and the left-hand part is the definition of s from Construction 5.19. Thus the outside commutes, and we see that s is a solution of e since $\widehat{a} = \widehat{[b, s]}$ holds by Corollary 5.21.

(2) We now prove that s is unique. Suppose that t is any solution of e. We will prove that

$$[b, t] = a, \tag{5.12}$$

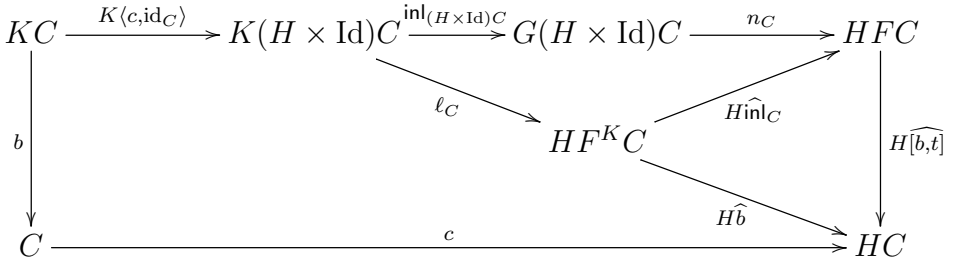
which implies the desired equation $s = t$.

In order to prove (5.12) we have to verify the commutativity of the following diagram (cf. Theorem 3.17):

$$\begin{array}{ccccc}
GC & \xrightarrow{G\langle c,\mathrm{id}_C\rangle} & G(H\times\mathrm{Id})C & \xrightarrow{n_C} & HFC \\
\downarrow{\scriptstyle [b,t]} & & & & \downarrow{\scriptstyle H\widehat{[b,t]}} \\
C & & \xrightarrow{\qquad\qquad c\qquad\qquad} & & HC
\end{array}$$

We verify this for the two coproduct components of $GC = KC + VC$ separately.

For the right-hand component we obtain the above diagram (5.11) with s replaced by t and a by $[b,t]$, which commutes since t is a solution of e. For the left-hand component we obtain the diagram below:



The big left-hand part commutes since $b : KC \to C$ is the ℓ-interpretation in C, the upper triangle commutes by the definition of n (see Construction 5.19), and for the right-hand triangle remove H and notice that

$$\widehat{[b,t]} \cdot \widehat{\mathsf{inl}}_C = \widehat{b}$$

by Remark 5.18(3).

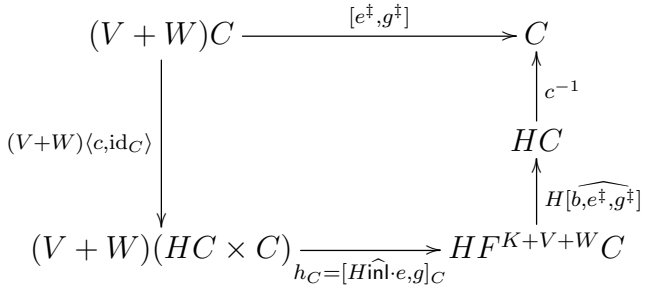
(3) To complete the proof we will show that $c^{-1} \cdot H\widehat{[b,s]} : HFC \to C$ is the structure of a CIA for HF. But this is a consequence of Theorem 4.7; indeed, recall that $[b,s]$ is the interpretation of the abstract GSOS rule $n : G(H \times \mathrm{Id}) \to HF$ in C, see Corollary 5.21. □

Remarks 5.22 (Compositionality of ℓ-RPS's). 1. Notice that the fact that the unique solution s of an ℓ-RPS extends the CIA structure on C means that the operations on C defined in this way may be part of recursive definitions according to the Corollaries 4.13 and 5.7.

2. In addition, we have a modularity principle for solutions of ℓ-RPS's—the operations provided by the unique solution $s : VC \to C$ may occur as givens in subsequent recursive definitions of operations (and we will make use of this feature in our applications in Section 5.2, e. g. to derive that operations defined by stream circuits may be used as building blocks in further stream circuits, see Remark 5.40 below). More precisely, we have seen in Construction 5.19 that the ℓ-RPS e gives rise to the abstract GSOS rule $n : G(H \times \mathrm{Id}) \to HF$, with the interpretation $[b,s] : GC \to C$. And by Theorem 5.16 every n-RPS $W(H \times \mathrm{Id}) \to HF^{G+W}$ has a unique solution $WC \to C$.

3. The modularity property from item (2) extends to a compositionality property of ℓ-RPS's as follows: let $e : V(H \times \mathrm{Id}) \to HF$ be an ℓ-RPS

and let $g : W(H \times \mathrm{Id}) \to HF^{K+V+W}$ be an n-RPS, where n arises from e as in Construction 5.19. Then we can compose e and g into the ℓ-RPS $h = [H\widehat{\mathsf{inl}} \cdot e, g] : (V+W)(H \times \mathrm{Id}) \to HF^{K+(V+W)}$. Its unique interpreted solution $h^\ddagger : (V+W)C \to C$ is the copairing $[e^\ddagger, g^\ddagger]$ of the solutions $e^\ddagger$ of e and $g^\ddagger$ of g as shown in the following diagram:



Its left-hand coproduct component commutes since $H[b, \widehat{e^\ddagger, g^\ddagger}] \cdot H\widehat{\mathsf{inl}}_C = H\widehat{[b, e^\ddagger]}$ (see Remark 5.18(3)) and since $e^\ddagger$ is a solution of e; the right-hand one commutes since $g^\ddagger$ is a solution of g. By the uniqueness result from Theorem 5.16 we have $h^\ddagger = [e^\ddagger, g^\ddagger]$.

Proposition 5.23. *Let $e_i : V_i(H \times \mathrm{Id}) \to HF^{K+V_i}$, $i = 1, 2$, be ℓ-RPS's. Then the CIA structure on C extended by the unique solutions $s_i : V_iC \to C$ of the e_i is independent of the order of extension.*

Remark 5.24. More precisely, we may first take $s_1 : V_1C \to C$ to obtain an extended CIA structure as in (5.10), and then take the solution of $s_2 : V_2C \to C$ in the new CIA, or vice versa. Either way, the resulting extended CIA structure is

$$HF^{K+V_1+V_2}C \xrightarrow{H\widehat{[b, s_1, s_2]}} HC \xrightarrow{c^{-1}} C\,. \tag{5.13}$$

Proof of Proposition 5.23. It is sufficient to prove that the CIA structure on C obtained by extending $k : HF^KC \to C$ from Theorem 4.7 first by s_1 and then by s_2 is (5.13).

So take s_1 and extend the CIA structure (C, k) to obtain the CIA $c^{-1} \cdot H\widehat{[b, s_1]} : HF^{K+V_1}C \to C$ (cf. (5.10)). Recall from the proof of Theorem 5.16 that this CIA structure is obtained as follows: one first forms the abstract GSOS rule

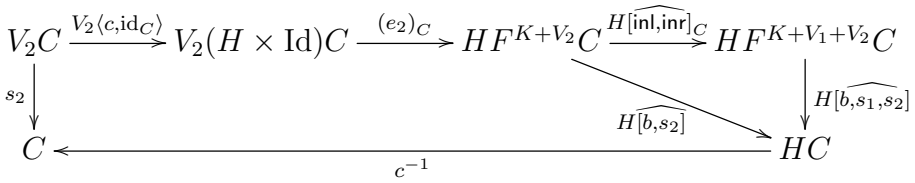
$$n = [H\widehat{\mathsf{inl}} \cdot \ell, e_1] : (K + V_1)(H \times \mathrm{Id}) \to HF^{K+V_1}$$

whose interpretation is $b' = [b, s_1] : (K + V_1)C \to C$, cf. Corollary 5.21, and then one applies Theorem 4.7.

Now we form the following n-RPS

$$V_2(H \times \mathrm{Id}) \xrightarrow{e_2} HF^{K+V_2} \xrightarrow{H\widehat{[\mathsf{inl},\mathsf{inr}]}} HF^{K+V_1+V_2}\,.$$

Its unique solution is easily seen to be s_2; indeed, consider the following diagram (and notice that the right-hand arrow is $H\widehat{[b', s_2]}$):



The left-hand part commutes since s_2 is the unique solution of e_2, and for the right-hand part we remove H and then precompose with $\kappa_C^{K+V_2}$ to obtain

$$\begin{aligned}
&\widehat{[b, s_1, s_2]} \cdot \widehat{[\mathsf{inl}, \mathsf{inr}]}_C \cdot \kappa_C^{K+V_2} \\
&= \widehat{[b, s_1, s_2]} \cdot \kappa_C^{K+V_1+V_2} \cdot [\mathsf{inl}, \mathsf{inr}]_C && \text{(by Remark 5.18(1))} \\
&= [b, s_1, s_2] \cdot [\mathsf{inl}, \mathsf{inr}]_C && \text{(by Remark 5.18(2))} \\
&= [b, s_2] \\
&= \widehat{[b, s_2]} \cdot \kappa_C^{K+V_2} && \text{(by Remark 5.18(2))}
\end{aligned}$$

The desired equality now follows since precomposition with κ^{K+V_2} is an isomorphism of categories, see Remark 5.18(2).

Finally, the CIA structure we obtain according to Theorem 5.16 is indeed (5.13). □

Before we turn to a "sandwiched" variant of ℓ-RPS's, we observe a fact which will be useful (also later in the concrete examples of Section 5.2):

Remark 5.25. Notice that we can always consider the algebraic operation obtained from $c^{-1} : HC \to C$ as a given operation in any ℓ-RPS for an abstract GSOS rule $\ell : K(H \times \mathrm{Id}) \to HF^K$. More precisely, we can assume that $K = K' + H$ and that the ℓ-interpretation $b : KC \to C$ has the form $b = [b', c^{-1}]$. Indeed, given any abstract GSOS rule $\ell' : K'(H \times \mathrm{Id}) \to HF^{K'}$ with ℓ'-interpretation b', we define the ℓ'-RPS

$$e = (\, H(H \times \mathrm{Id}) \xrightarrow{H\pi_0} HH \xrightarrow{H\mathsf{inr}} H(K' + H) \xrightarrow{H\kappa^{K'+H}} HF^{K'+H}\,)\,.$$

It is easy to verify that c^{-1} is its solution; and, as we see from the proof of Theorem 5.16, we obtain a new abstract GSOS rule $n : (K' + H)(H \times \mathrm{Id}) \to$

$HF^{K'+H}$ defined as in Construction 5.19 and having the n-interpretation $[b', c^{-1}]$.
Notice that in the special case of $K' = \emptyset$, i. e. starting with the empty abstract GSOS rule ℓ', n arises from $n' = \mathrm{id} : HH \to HH$ according to Remark 3.18. n' in turn is a trivial distributive law for which it was already shown in Section 4.4.2 of [Bar04] that it induces the operation c^{-1}.
Also notice that according to Proposition 5.23 we can do this construction at any step when defining operations, the result being always a GSOS rule $\ell : K(H \times \mathrm{Id}) \to HF^K$ with $K = K' + H$ and an ℓ-interpretation $[b', c^{-1}]$ containing the algebraic structure c^{-1}.

Sandwiched ℓ-RPS's

In Theorem 4.8 we proved that "sandwiched" flat equation morphisms $X \to F^K HF^K X + C$ have unique solutions (in some algebra k'). We will now prove a version of that theorem for ℓ-RPS's. The goal is to be able to uniquely solve specifications from a wider class.

Let us explain the idea with the help of an example—streams (cf. Section 4.2.1). Here we have $HX = \mathbb{R} \times X$ on Set. Suppose that K and V both are polynomial functors associated to a signature of givens and newly defined operations on the final H-coalgebra $C = \mathbb{R}^\omega$. The inverse of the final H-coalgebra structure $c = \langle \mathsf{hd}, \mathsf{tl} \rangle : \mathbb{R}^\omega \to \mathbb{R} \times \mathbb{R}^\omega$ yields the family of prefix operations $r.(-)$ prepending the number r to a stream. The format of an ℓ-RPS means that the new operations of type V are always defined by an equation with a prefix operation as a guard at its head, cf. for example the specification (5.5) of the shuffle product. This guard is sufficient to ensure a unique solution, however its position at the head of the term on the right-hand side is not necessary. For example, the shuffle product can also be specified by

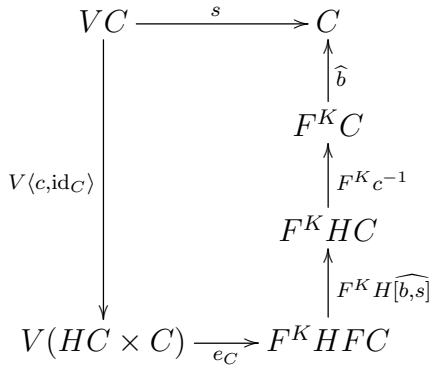
$$r.x \otimes s.y = (r \cdot s).(x \otimes s.y) + 0.(r.x \otimes y)\,.$$

We shall provide a result which makes this precise.

Definition 5.26. A *sandwiched recursive program scheme w. r. t. ℓ* (shortly, *sandwiched ℓ-RPS*) is a natural transformation $e : V(H \times \mathrm{Id}) \to F^K HF$.

An *interpreted solution* of e in C is a V-algebra $s : VC \to C$ such that

the diagram



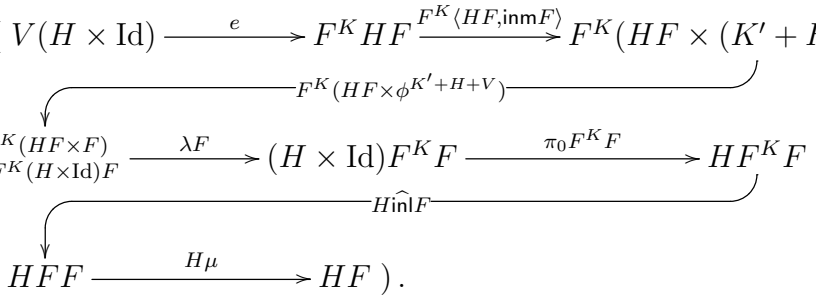
commutes.

Theorem 5.27. (Unique solutions of sandwiched ℓ-RPS's extend the initial CIA)**.** *For every sandwiched ℓ-RPS there exists a unique interpreted solution s in C. In addition, s extends the CIA structure on C, more precisely, the following is a CIA for FHF:*

$$FHFC \xrightarrow{FH\widehat{[b,s]}} FHC \xrightarrow{Fc^{-1}} FC \xrightarrow{\widehat{[b,s]}} C\,. \tag{5.14}$$

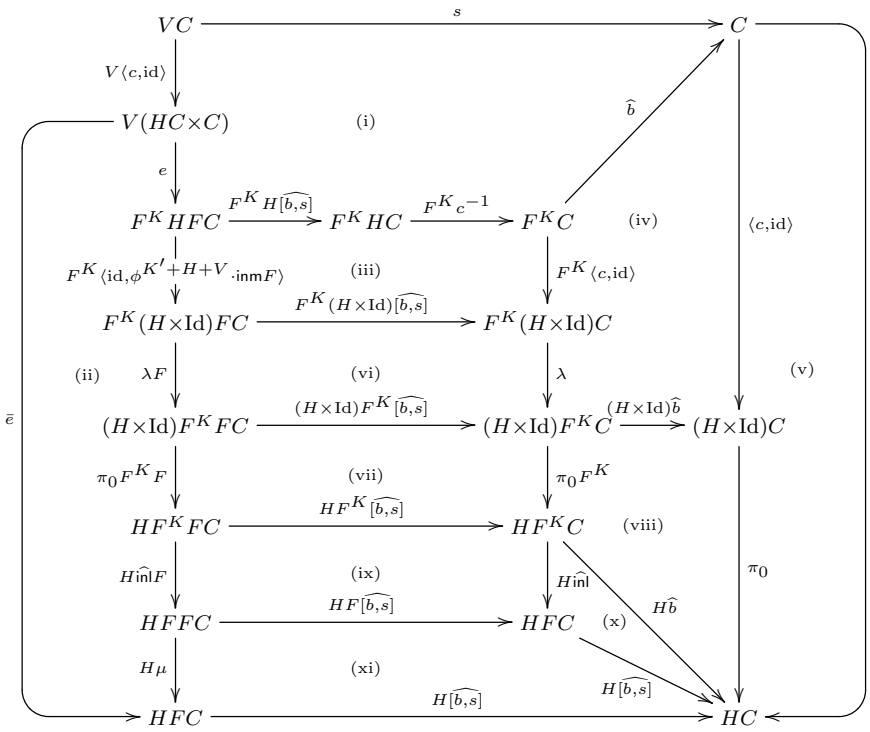
Proof. Recall from Remark 5.25 that we can assume $K = K' + H$ and that the ℓ-interpretation has the form $b = [b', c^{-1}]$. Furthermore recall from Lemma 3.14 that ℓ gives rise to a distributive law $\lambda : F^K(H \times \mathrm{Id}) \to (H \times \mathrm{Id})F^K$ of the free monad F^K over the cofree copointed functor $H \times \mathrm{Id}$.

Given a sandwiched ℓ-RPS $e : V(H \times \mathrm{Id}) \to F^K HF$, we form the (ordinary) ℓ-RPS $\bar{e} : V(H \times \mathrm{Id}) \to HF$ by defining



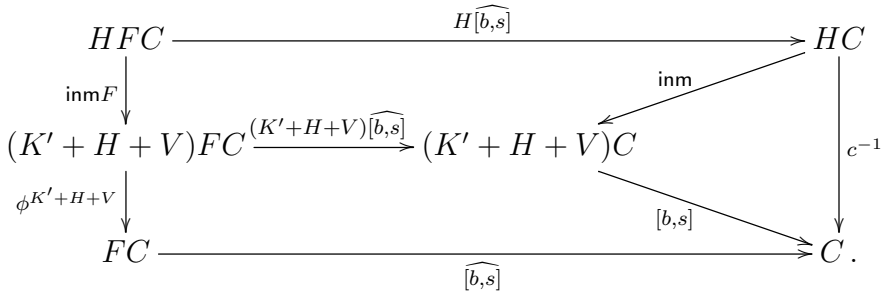
Then we verify that solutions of e and $\bar{e}$ are in one-to-one correspondence. Consider the following diagram (we drop the indices denoting components of

natural transformations):



(5.15)

We first show that all inner parts except part (i) commute: for part (ii) use the definition of $\bar{e}$, part (iv) is the commutative diagram in (3.2), part (v) trivially commutes, the parts (vi), (vii), (viii) and (ix) commute by the naturality of λ, $\pi_0 : H \times \mathrm{Id} \to H$ and $\widehat{\mathsf{inl}} : F^K \to F$, respectively, for part (x) use Remark 5.18(3), and part (xi) commutes since $\widehat{[b,s]}$ is the structure of an Eilenberg-Moore algebra for F. It remains to verify the commutativity of part (iii); we remove F^K and consider the product components separately: the left-hand component commutes using $c \cdot c^{-1} = \mathrm{id}_C$ and for the right-hand one we have the diagram



Its upper part commutes by the naturality of $\mathsf{inm} : H \to K' + H + V$, for the lower part recall from below Notation 3.4 the definition of $\widehat{[b,s]}$ as

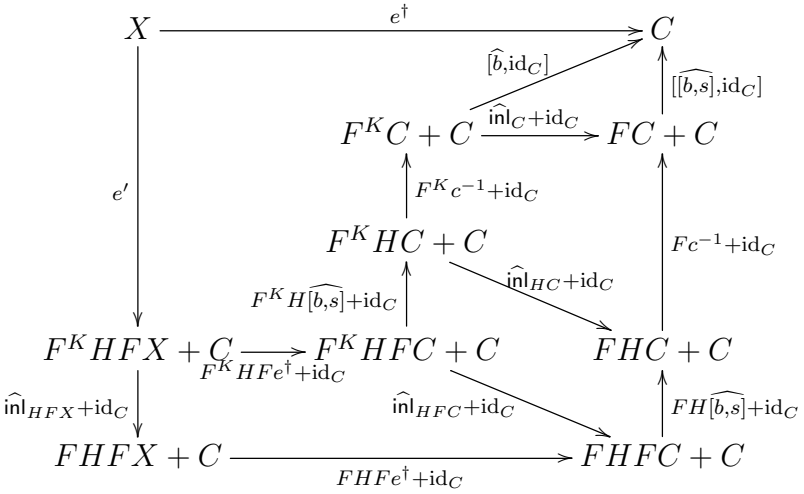
a homomorphism of algebras for $K' + H + V$, and the right-hand triangle commutes since $b = [b', c^{-1}]$, see Remark 5.25.

Now if s is a solution of $\bar{e}$, then the outside of diagram (5.15) commutes and therefore part (i) commutes, proving s to be a solution of e. Conversely, if s is a solution of e, part (i) commutes and then s is also a solution of $\bar{e}$. By Theorem 5.16, $\bar{e}$ has a unique solution; thus, e has a unique solution, too.

We still need to prove that (5.14) is the structure of a CIA for FHF. But this follows from Theorem 4.8. Indeed, from the ℓ-RPS $\bar{e}$ we form the abstract GSOS rule $n : G(H \times \mathrm{Id}) \to HF$ analogously as in Construction 5.19, and the n-interpretation is $[b, s]$ for the unique solution s of $\bar{e}$ (or, equivalently, of e). Now (5.14) is the structure k' from the statement of Theorem 4.8. □

For another example of a definition in the "sandwiched" format (besides the above mentioned definition of the shuffle product) see the definition of the CCS combinator alt in Section 5.2.3 below.

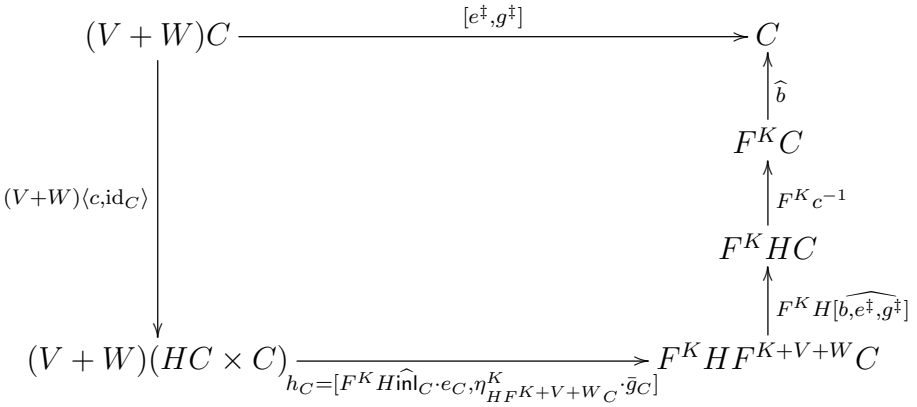
Remark 5.28. It is not difficult to see that also $\widehat{b} \cdot F^K c^{-1} \cdot F^K H\widehat{[b,s]} : F^K HFC \to C$ is the structure of a CIA: let $e' : X \to F^K HFX + C$ be a flat equation morphism and consider the following diagram:



All small inner parts commute by naturality of $\widehat{\mathsf{inl}}$ or by $\widehat{[b,s]} \cdot \widehat{\mathsf{inl}}_C = \widehat{b}$ (see Remark 5.18(3)). So we see that $e^\dagger$ is a solution of e' (i. e., the big inner part commutes) if and only if it is a solution of $(\widehat{\mathsf{inl}}_{HFX} + \mathrm{id}_C) \cdot e'$ (i. e., the outside commutes). But for the latter flat equation morphism we know that there is only one solution, and so we know this also for the former one.

Remark 5.29 (Compositionality of sandwiched ℓ-RPS's). Theorem 5.27 implies modularity of sandwiched ℓ-RPS's as follows: operations obtained as

solutions of sandwiched ℓ-RPS's can be used in subsequent recursive operation definitions, which will still have unique solutions. More precisely, we can apply Construction 5.19 to the ℓ-RPS $\bar{e}$ from the proof of Theorem 5.27 to obtain an abstract GSOS rule $n : G(H \times \mathrm{Id}) \to HF$, with the interpretation $[b, s] : GC \to C$ since the solution of $\bar{e}$ is the solution s of e. And by Theorem 5.27 every sandwiched n-RPS has a unique solution.
Moreover, this modularity property extends to a compositionality property of sandwiched ℓ-RPS's as follows: let $e : V(H \times \mathrm{Id}) \to F^K HF$ be a sandwiched ℓ-RPS and let $g : W(H \times \mathrm{Id}) \to FHF^{K+V+W}$ be a sandwiched n-RPS, where n is the abstract GSOS rule described above. Then we can compose e and g into the sandwiched ℓ-RPS $h = [F^K H\widehat{\mathsf{inl}} \cdot e, \eta^K HF^{K+V+W} \cdot \bar{g}] : (V + W)(H \times \mathrm{Id}) \to F^K HF^{K+V+W}$, where $\bar{g} : W(H \times \mathrm{Id}) \to HF^{K+V+W}$ is the n-RPS constructed from the sandwiched n-RPS g according to the proof of Theorem 5.27. The unique interpreted solution $h^\ddagger : (V + W)C \to C$ of h is the copairing $[e^\ddagger, g^\ddagger]$ of the solutions $e^\ddagger$ of e and $g^\ddagger$ of g as shown in the following diagram:



Its left-hand coproduct component commutes since $F^K H[\widehat{b, e^\ddagger, g^\ddagger}] \cdot F^K H\widehat{\mathsf{inl}}_C = F^K H[\widehat{b, e^\ddagger}]$ (see Remark 5.18(3)) and since $e^\ddagger$ is a solution of e. We show that the right-hand one also commutes: first it can be simplified using $\widehat{b} \cdot F^K c^{-1} \cdot F^K H[\widehat{b, e^\ddagger, g^\ddagger}] \cdot \eta^K_{HF^{K+V+W}C} = c^{-1} \cdot H[\widehat{b, e^\ddagger, g^\ddagger}]$ which holds by naturality of η^K and the property $\widehat{b} \cdot \eta^K_C = \mathrm{id}_C$ of the Eilenberg-Moore algebra $\widehat{b}$. To see that the simplified diagram commutes, we use that $g^\ddagger$ is a solution of the n-RPS $\bar{g}$, which follows from the proof of Theorem 5.27 since $g^\ddagger$ is the solution of the sandwiched n-RPS g, and that $[b, e^\ddagger]$ is the n-interpretation according to Corollary 5.21. Finally, by the uniqueness result from Theorem 5.27 we have $h^\ddagger = [e^\ddagger, g^\ddagger]$.

Proposition 5.30. *Let $e_i : V_i(H \times \mathrm{Id}) \to F^K H F^{K+V_i}$, $i = 1, 2$, be sandwiched ℓ-RPS's. Then the CIA structure on C extended by the unique solutions $s_i : V_i C \to C$ of the e_i is independent of the order of extension.*

Proof. The proof is similar to the proof of Proposition 5.23: it is sufficient to prove that the CIA structure on C obtained by extending $k' : F^K H F^K C \to C$ from Theorem 4.8 first by s_1 and then by s_2 is

$$\begin{array}{ccc} F^{K+V_1+V_2} H F^{K+V_1+V_2} C & \xrightarrow{F^{K+V_1+V_2} H\widehat{[b,s_1,s_2]}} & F^{K+V_1+V_2} HC \\ & & \downarrow{\scriptstyle F^{K+V_1+V_2} c^{-1}} \\ & & F^{K+V_1+V_2} C \xrightarrow{\widehat{[b,s_1,s_2]}} C\,. \end{array} \tag{5.16}$$

First the CIA structure (C, k') is extended by s_1 to obtain the CIA

$$\widehat{[b, s_1]} \cdot F^{K+V_1} c^{-1} \cdot F^{K+V_1} H\widehat{[b, s_1]} : F^{K+V_1} H F^{K+V_1} C \to C$$

(cf. (5.14)). Recall from the proof of Theorem 5.27 that the solution s_1 of e_1 equivalently is the solution of an (ordinary) ℓ-RPS $\bar{e}_1$ and that this CIA structure is obtained as follows: first one forms the abstract GSOS rule

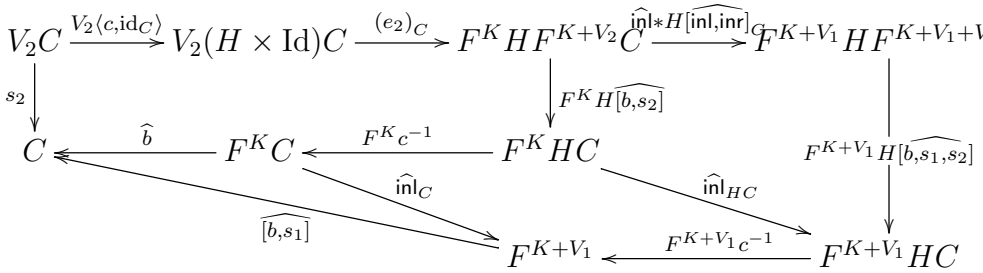
$$n = [H\widehat{\mathsf{inl}} \cdot \ell, \bar{e}_1] : (K + V_1)(H \times \mathrm{Id}) \to H F^{K+V_1}\,.$$

whose interpretation is $b' = [b, s_1] : (K + V_1)C \to C$, cf. Corollary 5.21, and then one applies Theorem 4.8.

Now we form the following sandwiched n-RPS

$$V_2(H \times \mathrm{Id}) \xrightarrow{e_2} F^K H F^{K+V_2} \xrightarrow{\widehat{\mathsf{inl}} * H\widehat{[\mathsf{inl},\mathsf{inr}]}} F^{K+V_1} H F^{K+V_1+V_2}\,.$$

Its unique solution is easily seen to be s_2; indeed, consider the following diagram (and notice that the right-hand arrow is $F^{K+V_1} H\widehat{[b', s_2]}$ and that $\widehat{[b, s_1]} = \widehat{b'}$):



The upper left-hand part commutes since s_2 is the unique solution of e_2, for the right-hand part we use parallel decomposition and use naturality of $\widehat{\mathsf{inl}}$

for the left-hand component; from the right-hand component we remove H and then precompose with $\kappa_C^{K+V_2}$ to obtain

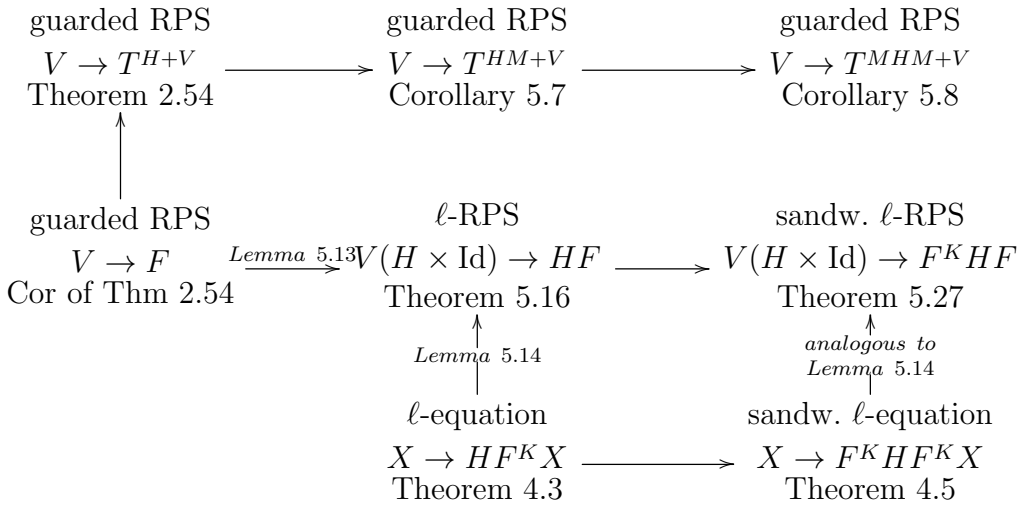
$$\begin{aligned}
&\widehat{[b, s_1, s_2]} \cdot \widehat{[\mathsf{inl}, \mathsf{inr}]}_C \cdot \kappa_C^{K+V_2} \\
&= \widehat{[b, s_1, s_2]} \cdot \kappa_C^{K+V_1+V_2} \cdot [\mathsf{inl}, \mathsf{inr}]_C && \text{(by Remark 5.18(1))} \\
&= [b, s_1, s_2] \cdot [\mathsf{inl}, \mathsf{inr}]_C && \text{(by Remark 5.18(2))} \\
&= [b, s_2] \\
&= \widehat{[b, s_2]} \cdot \kappa_C^{K+V_2} && \text{(by Remark 5.18(2))}
\end{aligned}$$

The desired equality now follows since precomposition with κ^{K+V_2} is an isomorphism of categories, see Remark 5.18(2). The lower left-hand triangle commutes by Remark 5.18(3), and the remaining lower part commutes due to naturality of $\widehat{\mathsf{inl}}$.

Finally, the CIA structure we obtain according to Theorem 5.27 is indeed (5.16). □

Remark 5.31. The format of a sandwiched ℓ-RPS, for which unique solutions in C exist by Theorem 5.27, comprises the format of an ℓ-RPS, for which unique solutions in C exist by Theorem 5.16: every ℓ-RPS $e : V(H \times \mathrm{Id}) \to HF$ can be considered as a sandwiched ℓ-RPS by taking $\eta^K HF \cdot e : V(H \times \mathrm{Id}) \to F^K HF$. It is easy to see that the respective solutions coincide (i. e., the solution is preserved by the construction).

Summarizing Remarks 5.9 and 5.31, we conclude this section with an overview of the different formats of recursive program schemes for the definition of operations on a final coalgebra we have seen (arrows point to more general formats):



Actually, the lowest line shows no formats of recursive program schemes, but formats of systems of recursive equations from Chapter 4. We added these to indicate that systems of recursive equations are a special case of recursive program schemes. However, our diagram does not capture all relations of this kind: one can e. g. prove that guarded equation morphisms (see Definition 4.11) are a special case of guarded RPS's (see Definition 2.52).
Also notice that the formats in the upper and middle entries in the middle and right-hand columns are not comparable: the upper entries allow for infinite terms, which the middle entries do not; and the middle entries allow for terms where given operations (corresponding to K) occur in subterms headed by new operations (corresponding to V) without an extra guarding operation (corresponding to H) occurring in the subterm, which the upper entries do not. Finally, we remark that only the upper row generalizes to arbitrary monads (or even pointed functors) M; for (sandwiched) ℓ-RPS's we need to work with free monads by Definitions 5.11 and 5.26.

5.2 Examples

In this section we go through the example final coalgebras from Section 4.2 once again:

- streams,
- infinite trees,
- CCS processes,
- formal languages and
- non-well-founded sets.

We show that known formats for the recursive definition of operations on streams (behavioral differential equations [Rut05a], stream circuits [Rut05b]) and on infinite trees (behavioral differential equations [SR10]) are instances of our ℓ-RPS format. We illustrate the modularity principle for ℓ-RPS's from Remark 5.22(2) with a step-by-step definition of several operations on formal languages. And we see in Remark 5.40 below that the compositionality principle from Remark 5.22(3) justifies our compositional approach to the construction of the stream circuit (I.2) from the introduction to Part I of this thesis. We give examples for the order independence of definitions from Proposition 5.23 (see Example 5.35 below) and for definitions by sandwiched ℓ-RPS's according to Theorem 5.27 (see Example 5.36 below). Along the

way, in this section we see lots of operations that can be defined using our formats; but we shall also see operation that are not definable with these (see Remark 5.43 below).

We continue writing F for the free monad F^{K+V} on $K+V$.

5.2.1 Streams

Recall the setting from Section 4.2.1 where $HX = \mathbb{R} \times X$ and $C = \mathbb{R}^\omega$.

Behavioral Differential Equations

Further recall from [Rut05a] that stream operations are defined by behavioral differential equations. For example, the componentwise addition of two streams σ and τ is specified by

$$(\sigma + \tau)_0 = \sigma_0 + \tau_0 \qquad\qquad (\sigma + \tau)' = (\sigma' + \tau')\,. \tag{5.17}$$

And the shuffle product (cf. (5.5)) is specified by

$$(\sigma \otimes \tau)_0 = \sigma_0 \cdot \tau_0 \qquad\qquad (\sigma \otimes \tau)' = (\sigma \otimes \tau' + \sigma' \otimes \tau)\,. \tag{5.18}$$

Rutten gives in [Rut05a] a general theorem for the existence of the solution of systems of behavioral differential equations. We will now recall this result and show that it is a special instance of our Theorem 5.16. For a system of behavioral differential equations one starts with the signature Σ of all the operations to be specified. One uses an infinite supply of variables, and for each variable x there is also a variable x' and a variable $x(0)$ (also written as x_0). For each operation symbol f from Σ one specifies

$$f(x_1, \ldots, x_n)_0 = h_f(x_1(0), \ldots, x_n(0)) \qquad\qquad f(x_1, \ldots, x_n)' = t_f\,, \tag{5.19}$$

where h_f denotes a function from $\mathbb{R}^n$ to $\mathbb{R}$ and t_f is a term built from operation symbols from Σ on variables x_i, x_i' and $x_i(0)$, $i = 1, \ldots, n$. Theorem A.1 of [Rut05a] asserts that for every f from Σ there exists a unique function $(\mathbb{R}^\omega)^n \to \mathbb{R}^\omega$ satisfying the equation (5.19) above.

We shall now show that a system such as (5.19) gives rise to an ℓ-RPS for a suitable abstract GSOS rule ℓ. To this end let $KX = \mathbb{R}$ be the constant functor and let

$$\ell = (\, K(H \times \mathrm{Id}) \xrightarrow{\ell'} HK \xrightarrow{H\kappa^K} HF^K \,)$$

with ℓ' given by $\ell'_X(r) = (r, 0)$. Then the ℓ-interpretation $b : \mathbb{R} \to C$ assigns to every $r \in \mathbb{R}$ the stream $b(r) = [r, 0, 0, \ldots]$.

Now given the system (5.19) let V be the polynomial functor associated to Σ, cf. Definition 2.2. Notice that $K + V$ is the polynomial functor of the signature Σ extended with a constant symbol r for every real number r. We translate the system (5.19) into an ℓ-RPS $e : V(H \times \mathrm{Id}) \to HF^{K+V} = HF$ as follows. For every f from Σ the corresponding component of e_X is defined by

$$e_X((r_1, x_1', x_1), \ldots, (r_n, x_n', x_n)) = (h_f(r_1, \ldots, r_n), \overline{t_f}),$$

where the term $\overline{t_f} \in FX$ is obtained by replacing in t_f all variables $x_i(0)$ by the constant r_i. Notice also, that here $h_f(r_1, \ldots, r_n)$ is a real number (the value of h_f at $(r_1, \ldots, r_n)$) whereas in (5.19) we have formal application of h_f to the variables $x_i(0)$.

It is now straightforward to verify that a solution of e in C corresponds precisely to a solution of the system (5.19). Thus, we obtain from Theorem 5.16 the

Corollary 5.32 (see also [Rut05a], Theorem A.1)**.** *Every system of behavioral differential equations has a unique solution.*

Example 5.33. For the system given by (5.17) and (5.18) we have $VX = X \times X + X \times X$ and e given componentwise as follows: for the $+$ component we have

$$e_X((r, x', x), (s, y', y)) = (r + s, x' + y') \tag{5.20}$$

and for the $\otimes$ component we have

$$e_X((r, x', x), (s, y', y)) = (r \cdot s, (x \otimes y') + (x' \otimes y)). \tag{5.21}$$

Observe that the systems (5.19) do not distinguish between given operations and newly defined ones. However, our result in Theorem 5.16 allows to make this distinction, and the modularity principle for solutions of ℓ-RPS's (cf. Remark 5.22(2)) means that operations specified by behavioral differential equations may be used in subsequent behavioral differential equations as given operations in the terms t_f from (5.19). This modularity principle for behavioral differential equations is a new result as well as the compositionality principle for behavioral differential equations which we obtain from Remark 5.22(3).

Example 5.34. Take $VX = X \times X$ and the ℓ-RPS e given by (5.20) whose solution is the operation of stream addition. As shown in Construction 5.19, ℓ and e yield an abstract GSOS rule $n : (K + V)(H \times \mathrm{Id}) \to HF$. Now let $V_1X = X \times X$. Then (5.21) yields an n-RPS e_1 whose solution is the shuffle product.

Next, we present an example illustrating Proposition 5.23.

Example 5.35. Continuing the previous example, consider the convolution product of streams specified by

$$(\sigma \times \tau)_0 = \sigma_0 \cdot \tau_0 \qquad (\sigma \times \tau)' = (\sigma' \times \tau + \sigma_0 \times \tau')\,,$$

see [Rut05a]. Let $V_2X = X \times X$ and let the n-RPS e_2 be given by

$$(e_2)_X((r, x', x), (s, y', y)) = (r \cdot s, (x' \times y) + (r \times y'))\,. \tag{5.22}$$

Notice that this illustrates why we introduced the constants r; in this way we are able to deal with σ_0 in the equation for $(\sigma \times \tau)'$. Then the unique solution of e_2 is the convolution product as expected. Proposition 5.23 asserts that the extended CIA structure for $HF^{K+V+V_1+V_2}$ on C does not depend on the order of taking the solution of (5.21) and (5.22)—either way this is given by the constants coming from $b : KC = \mathbb{R} \to C$ and the operations of stream addition as well as convolution and shuffle product.

Our results also allow to obtain unique solutions of specifications that go beyond behavioral differential equations, and we now provide one example.

Example 5.36. We give a sandwiched RPS w. r. t. ℓ defining the binary operation $\sigma \circ \tau$ assigning to two streams σ and τ the "undersampled" zipped stream

$$[0, \sigma_0, 0, \tau_0, 0, \sigma_1, 0, \tau_1, \dots]\,.$$

Indeed, let $KX = 1 + \mathbb{R} + X \times X + X \times X$ be the polynomial functor for the signature having a constant symbol c, a constant symbol for every $r \in \mathbb{R}$ and two binary symbols $+$ and $\times$. The abstract GSOS rule $\ell : K(H \times \mathrm{Id}) \to HF^K$ is given for each of the coproduct components separately by $\ell_X(*) = (0, 1)$ for the constant symbol c, $\ell_X(r) = (r, 0)$ for the constant symbols for every $r \in \mathbb{R}$, and similarly as in (5.20) and (5.22) for $+$ and $\times$. Now let $VX = X \times X$ (i. e., V corresponds to one binary operation $\circ$), and let $e : V(H \times \mathrm{Id}) \to F^K HF$ be the sandwiched ℓ-RPS given by

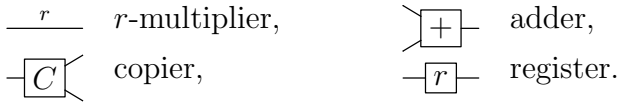
$$e_X((r, x', x), (s, y', y)) = \mathsf{c} \times (r, y \circ x')\,.$$

Then one easily verifies that the interpreted solution $C \times C \to C$ of e is the desired operation of "undersampled" zipping.

In general one can think of the right-hand sides of a sandwiched ℓ-RPS as terms of givens (represented by K) and newly defined operations (represented by V), where each newly defined operation must be guarded by a prefixing operation $r.-$.

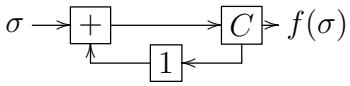
Stream Circuits

We now turn to another method to define streams—stream circuits [Rut05b], which are also called (signal) flow graphs in the literature. We shall demonstrate that specifications of streams by stream circuits arise as a special case of our results. Stream circuits are usually defined as pictorial compositions of the following basic stream circuits:



r-multiplier, adder, copier, register.

The r-multiplier multiplies all components of a stream by $r \in \mathbb{R}$, the adder performs componentwise addition, the copier yields two copies of a stream, and the register prepends $r \in \mathbb{R}$ to a stream σ to yield $r.\sigma$. The stream circuits are then built from the basic circuits by plugging wires together, and there may also be feedback (loops). A stream circuit is called *valid* if every loop passes through at least one register. For example the following picture shows a simple valid stream circuit (we direct the wires to illustrate the data flow from input to output):



(5.23)

It defines the unary function $f(\sigma) = (1 + \sigma_0, 1 + \sigma_0 + \sigma_1, 1 + \sigma_0 + \sigma_1 + \sigma_2, \dots)$ on streams.

For our treatment we shall consider the operations presented by r-multipliers, adders and registers as givens (copying will be implicit via variable sharing). Thus we work with the abstract GSOS rule ℓ from Section 4.2.1 that defines our given operations (here defined by behavioral differential equations):

$$\begin{aligned} (r\sigma)_0 &= r\sigma_0 & (r\sigma)' &= r\sigma' \\ (\sigma + \tau)_0 &= \sigma_0 + \tau_0 & (\sigma + \tau)' &= \sigma' + \tau' \\ (r.\sigma)_0 &= r & (r.\sigma)' &= \sigma \end{aligned}$$

We then obtain the ℓ-interpretation $b : \mathbb{R} \times C + C \times C + \mathbb{R} \times C \to C$.

It is well-known that every finite valid stream circuit with one input and one output defines a unique stream function, see [Rut05b]. Of course, a similar result holds for more than one input and output, and we present here a new proof of this result based on our Theorem 5.16.

Theorem 5.37 (cf. [Rut05b])**.** *Every finite valid stream circuit defines a unique stream function at every output.*

Proof. Let a finite valid stream circuit be given. We explain how to construct an ℓ-RPS from the circuit. Notice first that the wires in a circuit can be regarded as directed edges, cf. (5.23). We take for every register R in our circuit an operation symbol g_R and define its arity as the number of inputs that can be reached by following all possible paths from R backwards through the circuit. Similarly, we take for every output edge O of the circuit an operation symbol f_O with the arity obtained in the same way. Let Σ be the signature of all these symbols f_O and g_R, let V be the corresponding polynomial functor for Σ, and let Γ be the signature of the basic circuit operations. To give an ℓ-RPS $e : V(H \times \mathrm{Id}) \to HF$ it suffices to give a natural transformation $e' : VH \to HF$ and to define $e = e' \cdot V\pi_0$, where $\pi_0 : H \times \mathrm{Id} \to H$ is the projection. To obtain e', we give for each n-ary symbol s from Σ an assignment

$$s(r_1.x_1, \ldots, r_n.x_n) \mapsto (r, t)$$

where $r \in \mathbb{R}$ and t is a term built from symbols of $\Sigma + \Gamma$ using the variables $x_1, \ldots, x_n$. Notice that the arguments of s stand for generic elements (r_i, x_i) from HX for some set X and that the r may depend on r_i and t may contain the operation symbols $r_i.-$. We now show how to define the above assignment for each g_R. Suppose that R has the initial value r. Then

$$g_R(r_1.x_1, \ldots, r_n.x_n) \mapsto (r, t_R),$$

and we now explain how to obtain t_R: one follows every possible path in the circuit backwards that ends in R until

(i) an input edge corresponding to some argument $r_i.x_i$ is met, or

(ii) some register is met.

More precisely, we construct t_R as a $(\Sigma + \Gamma)$-tree: we follow the input edge of R backwards until we reach either the output wire of an r-multiplier, the output wire of an adder, an input wire of the whole circuit or the output wire of a register. For an r-multiplier or an adder we add a node to t_R labeled by the corresponding operation symbol and continue this process for each input node of the r-multiplier or adder constructing the corresponding subtrees of t_R. For an input wire corresponding to $r_i.x_i$ add a node labeled by the prefix operation $r_i.-$ and below that a leaf labeled by x_i; for a register S add the tree (of height 2) given by $g_S(r_{i_1}.x_{i_1}, \ldots, r_{i_k}.x_{i_k})$, where the $r_{i_j}.x_{i_j}$ correspond to those input wires of the circuit backwards reachable from the register S. Notice that these arguments of g_S form a subset of the arguments $\{ r_1.x_1, \ldots, r_n.x_n \}$ of g_R since every input that is backwards reachable from

S is also backwards reachable from R. Also notice that the copiers are ignored while forming t_R. Since the given circuit C is valid, we have indeed constructed a finite tree only, whence a term t_R.

We still need to define the assignment corresponding to e' for output symbols f_O:

$$f_O(r_1.x_1, \ldots, r_n.x_n) \mapsto (r, t_O)\,.$$

We first form the tree t'_O in essentially the same way as t_R for a register R with the difference that for every input wire and for every register we just insert an unlabeled leaf for the moment. To obtain r, label every leaf of t'_O corresponding to the input $r_i.x_i$ by r_i and every leaf corresponding to a register by its initial value; now evaluate the corresponding term to get r. In order to get t_O one replaces leaves of t'_O corresponding to inputs $r_i.x_i$ by x_i, and register leaves are replaced by the second components t_S from the right-hand sides of the equations for $g_S(r_{i_1}.x_{i_1}, \ldots, r_{i_k}.x_{i_k})$.

Finally, the unique solution of e yields a unique operation f_O on streams for every output O. By construction this is the operation computing the stream circuit. □

Remark 5.38. Suppose we are given a finite valid stream circuit where, in addition, every path from an input to an output passes through a register. Then the construction in the above proof would not need to refer to the behavior of the input $r_i.x_i$. That means that we could assume "structureless" inputs $x_1, \ldots, x_n$, and the above construction then even gives a guarded (ordinary) RPS $V \to T^{HF}$. Corollary 5.7 provides a unique solution of this RPS, and this result even allows for the unique solution of infinite valid stream circuits where every path from an input to an output passes through a register.

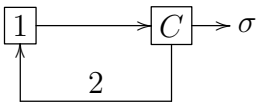
Example 5.39. The proof of Theorem 5.37 essentially gives a translation of an arbitrary finite valid stream circuit into an ℓ-RPS. We demonstrate this on the circuit given in (5.23) above. First we introduce for the output the function symbol f and for the register output the function symbol g. To determine their arities, we count the number of input wires which have a (directed) path to the register and the output, respectively. In both cases the arity is one. Now we must define $f(r.x)$ and $g(r.x)$ for an abstract input stream with head $r \in \mathbb{R}$. Each of these definitions is given by a pair (s, t) where $s \in \mathbb{R}$ and t is a term in the one variable x over operations corresponding to the basic circuits and f, g. We define

$$g(r.x) = (1, r.x + g(r.x)) \qquad f(r.x) = (r + 1, x + (r.x + g(r.x)))\,.$$

For $g(r.x)$ we take the value 1 of the register as first component, and the right-hand term is obtained as follows: we follow all paths from the register backwards until we find an input or a register. So we get a finite tree or, equivalently, the desired term. For $f(r.x)$ we first follow all paths to inputs and registers backwards to get the term $t' = x_I + x_R$, where x_I represents the input and x_R the register. For the first component of $f(r.x)$ we evaluate t' with the head r of the input and the initial value 1 of the register, and for the second component we replace in t' the input by x and the register by the second component of the right-hand side of the above definition of $g(r.x)$. The two equations above are easily seen to yield an ℓ-RPS $e : V(H \times \mathrm{Id}) \to HF$, where $V = \mathrm{Id} + \mathrm{Id}$ is the polynomial functor for the signature with two unary symbols f and g. The unique solution of e gives two unary operations (for f and g) on C, and the one for f is precisely the function computed by the circuit (5.23). By the modularity of stream circuits which will be explained next, we can use f (and also g) as "black boxes" in subsequent stream circuits.

Remark 5.40. The modularity principle for the unique solution of an ℓ-RPS we discussed in Remark 5.22(2) yields an important *modularity of stream circuits*: they can be used as building blocks as if they were basic operations in subsequent stream circuits. And Theorem 5.37 remains valid for the extended circuits.
Moreover, according to Remark 5.22(3) we have a *compositionality principle for stream circuits* in the sense that a stream circuit built in a modular way using nested building blocks defines the same operations as one that is built directly from the basic circuits. This principle was hinted at in [Rut05b], Exercise 4.14(b), and it verifies our claim from the introduction to Part I of this thesis that the stream circuit (I.2) indeed computes the stream squares.

Finally, let us consider the special case of *closed* stream circuits, i. e. stream circuits which have no inputs. For example the following picture shows a simple closed valid stream circuit:



(5.24)

This defines the stream $\sigma = [1, 2, 4, 8, \dots]$ of all powers of 2.

It is well-known (see [Rut05b]) that every closed finite valid stream circuit with one output defines a unique stream. This generalizes to stream circuits with several outputs:

Theorem 5.41 (cf. [Rut05b])**.** *Every closed finite valid stream circuit defines a unique stream at every output.*

We can also prove Theorem 5.41 by reducing stream circuits to our formats of systems of recursive equations or RPS's. We can choose one of the following methods: either we write down a direct construction of an ℓ-equation from a given closed finite valid stream circuit; then Theorem 5.41 follows from Theorem 4.3. Or we use Theorem 5.37 and see in the proof that the functor V of the constructed ℓ-RPS is a constant functor; thus we can apply Lemma 5.14 to break this ℓ-RPS down to an ℓ-equation and use Theorem 4.3 again.

5.2.2 Infinite Trees

Recall the setting of Section 4.2.2 where $HX = X \times \mathbb{R} \times X$ and where C is the set of all (complete) infinite binary trees with node labels in $\mathbb{R}$.

Silva and Rutten [SR10] define single trees (constants) and operations on trees by behavioral differential equations. For example,

$$\mathsf{pi}(\varepsilon) = \pi \qquad \begin{aligned} \mathsf{pi}_L &= \mathsf{pi} \\ \mathsf{pi}_R &= \mathsf{pi} \end{aligned}$$

specifies the tree pi with every node labeled by the number π. For every real number r we have the constant $[r]$ specified by

$$[r](\varepsilon) = r \qquad \begin{aligned} [r]_L &= [0] \\ [r]_R &= [0]\,. \end{aligned}$$

This generalizes to the definition of operations on infinite trees by behavioral differential equations, and Silva and Rutten [SR10] proved a theorem asserting that these have indeed unique solutions. Here we shall show that, similar to Corollary 5.32 for streams, we obtain that theorem as a special instance of our Theorem 5.16.

Let us start by giving an example of an operation on infinite trees defined by behavioral differential equations: the nodewise addition of numbers stored in the nodes of the trees t and s (cf. the abstract GSOS rule ℓ from Section 4.2.2) is defined by

$$(t+s)(\varepsilon) = t(\varepsilon) + s(\varepsilon) \qquad \begin{aligned} (t+s)_L &= t_L + s_L \\ (t+s)_R &= t_R + s_R\,. \end{aligned}$$

See [SR10] for further and more exciting examples.

In general a system of behavioral differential equations is specified as follows. Let W be an infinite set of syntactic variables. For every $x \in W$ we have the notational variants x_L, x_R and also $x(\varepsilon)$. Furthermore, let Σ be a

signature of operations to be specified. For each operation symbol f from Σ of arity n we provide equations of the form

$$\begin{array}{c|c} \text{initial value} & \text{differential equations} \\ \hline (f(x_1,\ldots,x_n))(\varepsilon) = c_f(x_1(\varepsilon),\ldots,x_n(\varepsilon)) & \begin{array}{c} f(x_1,\ldots,x_n)_L = t_1 \\ f(x_1,\ldots,x_n)_R = t_2 \end{array} \end{array} \tag{5.25}$$

where c_f denotes a function $\mathbb{R}^n \to \mathbb{R}$ and t_1 and t_2 are Σ-terms on the variables $x_1,\ldots,x_n$ and their three notational variants.

From Theorem 5.16 we obtain the

Corollary 5.42 (see also [SR10], Theorem 2)**.** *Every system* (5.25) *of behavioral differential equations has a unique solution, i. e., for every f from Σ there exists a unique function $f : C^n \to C$ satisfying* (5.25)*.*

Proof. Let $KX = \mathbb{R}$ be the constant functor, and let the abstract GSOS rule ℓ be given, according to Remark 3.18, by $\ell' : K(H \times \mathrm{Id}) \to HK$ defined by $\ell'_X(r) = (0, r, 0)$. Then the ℓ-interpretation is $b : \mathbb{R} \to C$ with $b(r) = [r]$. Now every system (5.25) gives an ℓ-RPS $e : V(H \times \mathrm{Id}) \to HF$ as follows: let V be the polynomial functor associated to Σ and let e be given on each component corresponding to f from Σ by

$$e_X(((x_1)_L, r_1, (x_1)_R, x_1), \ldots, ((x_n)_L, r_n, (x_n)_R, x_n)) = (\overline{t_1}, c_f(r_1,\ldots,r_n), \overline{t_2})\,,$$

where $\overline{t_i}$ is obtained from t_i by replacing each $x_i(\varepsilon)$ by the corresponding constant r_i. By construction, the solutions of e in C correspond precisely to solutions of (5.25); thus, Corollary 5.42 follows from Theorem 5.16. □

In addition, we have again a modularity principle (cf. Remark 5.22(2)): operations specified by behavioral differential equations may be used as givens in subsequent behavioral differential equations. And again, this extends to a compositionality principle for behavioral differential equations for infinite trees, see Remark 5.22(3).

5.2.3 CCS Processes

Recall the setting from Section 4.2.3 where $HX = \mathcal{P}_\kappa(A \times X)$ and where C is the set of all strongly extensional κ-branching trees with edges labeled in A, and these trees can be considered as the CCS agents or CCS processes modulo strong bisimilarity.

Suppose we want to define the binary combinator "alt" which performs alternation of two processes. For its definition we shall need another binary combinator, sequential composition of two processes (denoted by the infix

";"). Here we suppose that the latter combinator is already included in our basic calculus—more precisely, we could have added a sixth coproduct component $X \times X$ to K for sequential composition in the definition of the abstract GSOS rule ℓ in Section 4.2.3 and could have completed ℓ by

$$\ell_X(S_1, x_1, S_2, x_2) = \begin{cases} \{(a, x; x_2) \mid (a, x) \in S_1\} & \text{if } S_1 \neq \emptyset \\ S_2 & \text{if } S_1 = \emptyset \end{cases}$$

for this coproduct component. This gives indeed the desired combinator for sequential composition as part of the ℓ-interpretation. Observe in particular that Theorem 4.29 still holds for the calculus including this sixth combinator. Now for this extended ℓ we give a sandwiched ℓ-RPS $e : V(H \times \mathrm{Id}) \to F^K HF$, where $VX = X \times X$, in order to define the combinator alt:

$$e_X(S_1, x_1, S_2, x_2) = \begin{cases} S_1; \{(a, x; \mathsf{alt}(x_1, x_2)) \mid (a, x) \in S_2\} & \text{if } S_2 \neq \emptyset \\ \{(a, x; \mathsf{alt}(x_2, x_1)) \mid (a, x) \in S_1\} & \text{if } S_1 \neq \emptyset, S_2 = \emptyset \\ \emptyset & \text{if } S_1 = S_2 = \emptyset\,. \end{cases}$$

Notice that the term in the first line of this definition does not lie in HF, so Theorem 5.16 cannot be applied. But it does lie in $F^K HF$, so Theorem 5.27 tells us that e has a unique solution $s : C \times C \to C$. It is not difficult to see that this is alternation of processes indeed. Furthermore, Theorem 5.27 tells us that s extends the CIA structure from Theorem 4.8 applied to the extended abstract GSOS rule ℓ. This means that Theorem 4.29 remains true for the calculus extended by sequential composition and alternation of processes, without further work.

Finally, suppose we want to define two unary combinators op_1 and op_2 by the SOS rule (cf. (4.10) in Section 4.2.3)

$$\frac{E \xrightarrow{a} F}{\mathsf{op}_1(E) \xrightarrow{a} F|\mathsf{op}_2(F + E) \quad \mathsf{op}_2(E) \xrightarrow{a} F + \mathsf{op}_1(F|E)}\,.$$

Then Theorem 5.16 tells us that this rule uniquely determines the two combinators. Indeed, we translate the rule into an ℓ-RPS: let $V = \mathrm{Id} + \mathrm{Id}$ (two unary combinators are specified) and let $e : V(H \times \mathrm{Id}) \to HF$ be given by

$$e(S, x) = \{(a, y|\mathsf{op}_2(y + x)) \mid (a, y) \in S\}$$

on the first component and

$$e(S, x) = \{(a, y + \mathsf{op}_1(y|x)) \mid (a, y) \in S\}$$

on the second one. The unique solution of e gives us two new unary combinators on C extending its CIA structure. Again this means that Theorem 4.29 remains true for the extended calculus, without further work.

5.2.4 Formal Languages

Recall the setting of Section 4.2.4 where $HX = X^A \times 2$ and $C = \mathcal{P}(A^*)$.

We shall now show how various operations on formal languages can be defined in a modular way using Theorem 5.16. It is well-known that such operations can be defined as interpretations of one abstract GSOS rule (or distributive law) in C, see e. g. the paper [Jac06a] of Jacobs. However, this paper does not explain why one may define these operations in a step-by-step fashion by subsequent recursive definitions. This is the added value of Theorem 5.16.

We start with the functor $K_0 = C_\emptyset$ (that means, we start from scratch with no given operations) and with the obvious $\ell_0 : C_\emptyset(H \times \mathrm{Id}) \to HF^{C_\emptyset} = H$ given by the empty maps. The corresponding interpretation is the empty map $b : \emptyset \to C$, and $\widehat{b}$ is then the identity on C. Thus, the CIA structure for HF^{K_0} on C given by Theorem 4.7 is simply the initial CIA (C, c^{-1}) for H. At each subsequent step we are given a functor K_i and an abstract GSOS rule $\ell_i : K_i(H \times \mathrm{Id}) \to HF^{K_i}$ with its interpretation $b_i : K_iC \to C$. We then give an ℓ_i-RPS $e_i : V_i(H \times \mathrm{Id}) \to HF^{K_i+V_i}$, and its unique solution $s_i : V_iC \to C$ extends the CIA structure as follows: let $K_{i+1} = K_i + V_i$ and let $\ell_{i+1} = [H\widehat{\mathsf{inl}} \cdot \ell_i, e_i] : K_{i+1}(H \times \mathrm{Id}) \to HF^{K_{i+1}}$, where $\widehat{\mathsf{inl}} : F^{K_i} \to F^{K_{i+1}}$ is the monad morphism induced by $\mathsf{inl} : K_i \to K_{i+1}$ (cf. Notation 5.17(1)). By induction it is easy to see that the ℓ_{i+1}-interpretation is $b_{i+1} = [s_j]_{j=0,\ldots,i} : K_{i+1}C \to C$ (cf. Corollary 5.21). And this gives an extended CIA $c^{-1} \cdot H\widehat{b}_{i+1} : HF^{K_{i+1}}C \to C$ by Theorem 4.7.

As a first step we define constants in C for $\emptyset$, $\{\varepsilon\}$, and $\{a\}$, $a \in A$, as solutions of an ℓ_0-RPS. We express this as an ℓ_0-RPS as follows: take the functor $V_0X = 1 + 1 + A$ corresponding to the above constants. We define $e_0 : V_0(H \times \mathrm{Id}) \to HF^{K_0+V_0} = HF^{V_0}$ componentwise. We write for every set X, $\emptyset$ for $\mathsf{inj}_1(*) \in V_0X$ and ε for $\mathsf{inj}_2(*) \in V_0X$. Then $(e_0)_X$ is given by the assignments

$$\begin{aligned} &\emptyset \mapsto ((\emptyset)_{a\in A}, 0)\\ &\varepsilon \mapsto ((\emptyset)_{a\in A}, 1)\\ &b \mapsto ((t_a), 0) \qquad\qquad \text{where } t_a = \begin{cases} \varepsilon & \text{if } a = b \\ \emptyset & \text{if } a \neq b\,, \end{cases} \end{aligned}$$

cf. Section 4.3.3. It is now straightforward to check that the unique solution s_0 of e_0 yields the desired constants in C extending the CIA structure.

Next we add the operations of union, intersection and language complement to the CIA structure. Let $K_1 = K_0 + V_0$ and let $\ell_1 = [H\widehat{\mathsf{inl}} \cdot \ell_0, e_0]$ as above with interpretation $b_1 = s_0$. Let $V_1X = X \times X + X \times X + X$ be

the polynomial functor corresponding to two binary symbols $\cup$ and $\cap$ and one unary one $\overline{(-)}$. We give the ℓ_1-RPS $e_1 : V_1(H \times \mathrm{Id}) \to HF^{K_1+V_1}$ componentwise in the form of the three assignments in (5.26) below. We write $((x_a), j, x)$ for elements of $HX \times X$, where (x_a) is an A-tuple, i. e., an element of X^A. We also write elements of $V_1 Z$, $Z = HX \times X$, as flat terms $z_1 \cup z_2$, $z_1 \cap z_2$ and $\overline{z}$ for the three components:

$$\begin{aligned} ((x_a), j, x) \cup ((y_a), k, y) &\mapsto ((x_a \cup y_a), j \vee k) \\ ((x_a), j, x) \cap ((y_a), k, y) &\mapsto ((x_a \cap y_a), j \wedge k) \\ \overline{((x_a), j, x)} &\mapsto ((\overline{x_a}), \neg j) \end{aligned} \tag{5.26}$$

where $\vee$, $\wedge$ and $\neg$ are the evident operations on $2 = \{0, 1\}$, cf. Section 4.3.3. The corresponding unique solution $s_1 : V_1 C \to C$ is easily checked to provide the desired operations extending the CIA structure on C.

The next step adds concatenation to the CIA structure on C. For this let $V_2 X = X \times X$ and e_2 is given by the assignment

$$((x_a), j, x) \cdot ((y_a), k, y) \mapsto ((t_a), j \wedge k) \qquad \text{where } t_a = \begin{cases} (x_a \cdot y) \cup y_a & \text{if } j = 1 \\ x_a \cdot y & \text{else}\,. \end{cases}$$

Its unique solution $s_2 : C \times C \to C$ is the concatenation operation, cf. the last equation in (4.11).

As the final step we add the Kleene star operation by taking $V_3 X = X$ and e_3 given by

$$e_3((x_a), j, x) = ((x_a \cdot x^*), 1)\,,$$

cf. Section 4.3.3. Notice that this definition makes use of concatenation which was a solution at the previous stage and concatenation makes use of union which was a solution at stage 1.

Remark 5.43. Most of the common operations on formal languages like union, concatenation or Kleene star are definable by ℓ-RPS's as we have seen above. This is also true for the following further examples of operations:

- prefixing $a.L = \{aw \mid w \in L\}$, $a \in A$, cf. Section 4.3.3;
- the operation given by $c^{-1} : C^A \times 2 \to C$ (see Remark 5.25)

$$((L_a), j) \mapsto \begin{cases} \bigcup_{a \in A} a.L_a & \text{if } j = 0 \\ \bigcup_{a \in A} a.L_a \cup \{\varepsilon\} & \text{else}\,; \end{cases}$$

- $\mathsf{zip}(L_1, L_2) = \bigcup_{w_1 \in L_1, w_2 \in L_2} \mathsf{zip}(w_1, w_2)$ where $\mathsf{zip}(w_1, w_2)$ is the usual operation zipping the words w_1 and w_2;

- $\mathsf{shuffle}(L_1, L_2) = \bigcup_{w_1 \in L_1, w_2 \in L_2} \mathsf{shuffle}(w_1, w_2)$ where $\mathsf{shuffle}(w_1, w_2)$ is the usual operation merging the words w_1 and w_2.

We leave it to the reader to work out the details. Notice, however, that there exist operations that cannot be defined by ℓ-RPS's (or abstract GSOS rules). An example is the language derivative $L^a = \{w \mid aw \in L\}$ for some $a \in A$. Indeed, if this was definable by an ℓ-RPS, Theorem 5.16 would yield a CIA structure on C for the functor HF^{K+V}, where $VX = X$ corresponds to the unary operation $(-)^a$. For a term t in $F^{K+V}X$, we shall use the notation $a.t$ as a shortcut for $((t_b), 0) \in HF^{K+V}X$ for the A-tuple (t_b) with $t_a = t$ and $t_b = \emptyset$ for every $b \in A \setminus \{a\}$. Thus, for $X = \{x\}$ the flat equation morphism $e : X \to HF^{K+V}X + C$ given by $e(x) = a.x^a$ would have a unique solution. But this is clearly not the case: every formal language L whose words all start with a gives a solution $e^\dagger(x) = L$ of the flat equation morphism e.

5.2.5 Non-well-founded Sets

Recall the setting from Section 4.2.5 where $HX = \mathcal{P}X$ is an endofunctor on Class and where $C = V$ is the class of all non-well-founded sets. The results of Section 5.1.2 hold true for Class since every endofunctor of Class has final coalgebras and free algebras, see [AMV04].

Remark 5.44. Recall the (first) abstract GSOS rule ℓ defined in Section 4.2.5. We now can obtain the various operations on V defined by ℓ in a step-by-step fashion starting with b_4 and b_5 and then defining b_1, b_2, b_3 by successive applications of Theorem 5.16 as we did in the previous section on formal languages.

Now one may uniquely solve recursive function definitions such as

$$g(x) = \{g(\mathcal{P}(x)) \times x, x\}\,.$$

Indeed, for $W = \mathrm{Id}$ this recursive equation yields an ℓ-RPS $e : W(\mathcal{P} \times \mathrm{Id}) \to \mathcal{P}F^{K+W}$ whose unique solution given by Theorem 5.16 is a function $g_V : V \to V$ behaving as specified.

5.3 Related Work

A category theoretic notion of recursive program schemes has been developed by Ghani, Lüth and De Marchi [GLM03] and by Milius and Moss [MM06], the latter being the paper on which our Section 5.1.1 builds. We also showed in Lemma 5.13 how our ℓ-RPS's are related to this notion of an "ordinary"

RPS. But the ℓ-RPS's and the techniques used in this context have more in common with the abstract GSOS rules of Bartels [Bar03, Bar04]: they can be viewed as "half an abstract GSOS rule", namely for the second coproduct half of $K + V$, cf. Construction 5.19. Whereas Bartels develops a whole framework for different variants of distributive laws, the distributive laws corresponding to abstract GSOS rules have been recognized earlier as an important definition format by Turi and Plotkin [TP97] and the subsequent work of Lenisa, Power and Watanabe [LPW00].

Modularity (cf. Remarks 5.5, 5.22(2) and 5.29) in mathematical operational semantics has been studied before in [Pow03, LPW04]. These papers show how to combine two different specifications of operations over the same behavior by performing constructions at the level of distributive laws. This gives an abstract explanation of adding operations to a process calculus. At the heart of our proof of Theorem 5.16 lies a construction similar to the combination of two distributive laws that arises by taking the coproduct of the corresponding monads. However, while in the coproduct construction of [Pow03, LPW04] the two distributive laws are independent of each other, our case is different because the operations specified by the second distributive law interact with the operations specified by the first one. The same difference applies to the work of Jaskelioff, Ghani and Hutton [JGH11], who presented modular syntax in the programming language Haskell by taking coproducts of free monads.

Much less related to our work is the work of Kick and Power [KP04, KPS06] who show how to combine distributive laws of one monad over two sorts of behavior possibly interacting with one another. Technically, they compose coalgebra functors (actually their free comonads) using products instead of algebra functors using coproducts.

Concerning our Examples in Section 5.2, behavioral differential equations for streams have been developed by Rutten [Rut05a, Rut05b]; and behavioral differential equations for infinite trees by Silva and Rutten [SR07, SR10]. It is clear from these two similar examples of recursive definition formats that the idea to work with an abstract coalgebra functor H is not new. However, the advantages of our approach are that (a) it provides a generic format from which concrete formats for different functors H are easily derived without further proofs that definitions in these concrete formats have unique solutions, and (b) it gives a mathematical foundation for the modular and compositional use of behavioral differential equations. The format of stream circuits is also due to Rutten [Rut08]; again the advantages (a) and (b) apply, showing moreover that one can generalize the results to open valid stream circuits. The definition format for CCS agents was invented by Milner [Mil89]. For formal languages (see e. g. [HMU07]) and non-well-founded

sets (see [Acz88, BM96]) we are not aware of general definition formats for operations on them. The examples from Section 5.2.5 were provided by Moss.

Most of the results from Section 5.1 can be found in our joint paper [MMS13] with Milius and Moss (the special issue version of the conference paper [MMS10]). Exceptions are Lemmata 5.13, 5.14 and 5.15 and Proposition 5.30 which were not proved there. The compositionality results from Remarks 5.5, 5.22 and 5.29 as well as Lemma 5.6 are new here. Finally, nearly all of the examples from Section 5.2 can also be found in [MMS13].

Part II

Effectful Computations in Recursive Specifications

This second part of the thesis is concerned with *computational effects* in recursive specifications. Next we illustrate what we understand by "effects" on the example of nondeterminism. Recall the terminology introduced in Section 1.1.

Nondeterministic Recursive Equations

A *nondeterministic computation* is a computation whose result is taken from a set of possible results but cannot be foreseen: we say that the result is *chosen nondeterministically*. Nondeterministic computations serve as an abstraction in modeling systems: for example, modeling a board game we abstract away from a concrete play with concrete players and next moves, and only describe a play abstractly by giving a set of possible next moves, i. e. we view the decision of a player for a next move as a nondeterministic computation.

As an example for a nondeterministic computation consider the function next which computes the next position of a knight on the chessboard. Recall that a chessboard is an (8×8)-matrix of squares and a knight's move is one square in horizontal and two squares in vertical direction or vice versa. Thus there are several possible next positions from which the player may choose.

Our function next acts on the set $A = \{CR \mid a \leq C \leq h \text{ and } 1 \leq R \leq 8\}$ of squares $a1$ to $h8$ on the chessboard. We model nondeterminism by giving the set of all possible choices, i. e. we have $\mathsf{next} : A \to \mathcal{P}A$ given by

$$\mathsf{next}(C_0R_0) = \{CR \in A \mid 1 \leq |C{-}C_0|, |R{-}R_0| \leq 2 \text{ and } |C{-}C_0| \neq |R{-}R_0|\}\,.$$

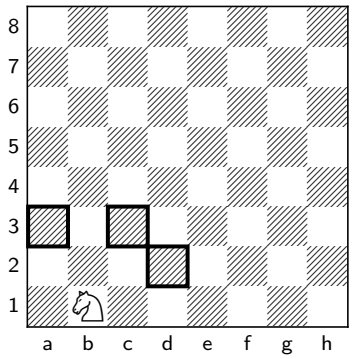


Figure II.1: knight's moves

For example, for the initial position $b1$ of the left white knight the function yields $\mathsf{next}(b1) = \{a3, c3, d2\}$ as shown in Figure II.1.

Now consider the variant $\mathsf{next}^+ : A \to \mathcal{P}A$ of the function next where only forward moves (to rows with greater row number) are permitted, i. e. $\mathsf{next}^+(C_0R_0) = \{CR \in \mathsf{next}(C_0R_0) \mid R - R_0 > 0\}$. We would like to model the set $K \subseteq A$ of all squares which can be reached by forward moves of the left white knight from its initial position. It is shown in Figure II.2.

K should satisfy the equation $K = \{b1, \mathsf{next}^+(K)\}$ where next^+ applied to a subset S of A denotes the extension of next^+ to sets by applying it to each element and taking the union of all resulting sets; the set braces with elements and sets inside mean the "flattened" set which contains all elements and all elements from the sets. Indeed, this equation states that K contains the initial position $b1$ (reachable in zero forward moves) and all the squares reachable in one forward move from any square in K. As we shall see in Example 7.52(3) below, this equation suffices to characterize the set K—in fact, K is the unique solution of the *nondeterministic recursive equation* $x = \{b1, \mathsf{op}(x)\}$ in the *nondeterministic algebra* (A, next^+). Moreover, we shall give a characterization of the algebras in which all systems of flat nondeterministic recursive equations have a unique solution in Theorem 7.50.

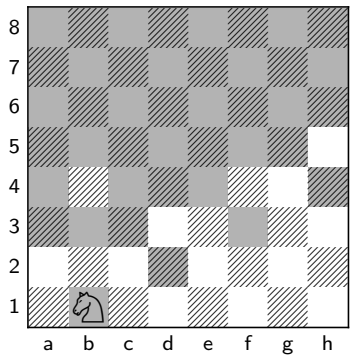


Figure II.2: the set K

Nondeterministic Recursive Program Schemes

One may also investigate *nondeterministic recursive program schemes*—in fact, this was the topic of several publications in the 1970's and 80's (see e. g. [Bou80, AN80, Poi82]). More recently, a category theoretic semantics for deterministic recursive program schemes has been developed by Ghani et al. [GLM03] and by Milius and Moss [MM06]. However, no category theoretic semantics for nondeterministic schemes has been presented so far. We shall fill this gap, which turns out to be a technically challenging task.

Let us come back to our example above, but instead of the constant set K we would now like to characterize the function $\mathsf{knight} : A \to \mathcal{P}A$ which takes an arbitrary initial position of a knight on the chessboard and returns all the squares that can be reached by forward moves. It should satisfy the equation $\mathsf{knight}(a) = \{a, \mathsf{next}^+(\mathsf{knight}(a))\}$ for every $a \in A$, where next^+ applied to a subset S of A and the set braces are interpreted as in the above equation for K. Indeed, this equation states that $\mathsf{knight}(a)$ contains the initial position a and all squares reachable in one forward move from any square in $\mathsf{knight}(a)$. This equation gives rise to the nondeterministic recursive program scheme (written in the classical notation using or)

$$\phi(x) = x \ \mathsf{or} \ \mathsf{op}(\phi(x))\,. \tag{II.1}$$

Here ϕ is a *new function symbol* of arity one, op a *given function symbol* of

arity one and x a *variable*. Interpreting the scheme in the algebra (A, next^+) and the operation symbol or as nondeterministic choice we see that the function knight is an *interpreted solution* of the scheme.

However, we shall not consider interpreted solutions of nondeterministic schemes in this thesis but only the more basic notion of an *uninterpreted solution*. An uninterpreted solution of a nondeterministic scheme assigns to every new function symbol a set of possibly infinite trees with nodes labeled by the given function symbols or the variables, which solves the scheme. For our example scheme (II.1) the following infinite set

$$\left\{ x,\ \mathsf{op}(x),\ \mathsf{op}(\mathsf{op}(x)),\ \mathsf{op}(\mathsf{op}(\mathsf{op}(x))),\ \ldots,\ \mathsf{op}^{\omega} \right\} \tag{II.2}$$

of trees, where the right-hand tree is infinite, is an uninterpreted solution: substituting this set for $\phi(x)$ on the right-hand side of (II.1), extending op to sets by applying it to each element, and interpreting the operation symbol or as the nondeterministic choice, we get this set back. But removing the infinite tree from this set also gives a solution—unlike for deterministic recursive program schemes we must make a decision when giving a semantics to nondeterministic schemes. In our main result about nondeterministic schemes (Theorem 8.60 below) we prove that there always is a greatest uninterpreted solution, and it can be chosen canonically.

A Framework for Several Computational Effects

We shall not restrict ourselves to nondeterminism—our category theoretical approach gives us a framework in which different computational effects can be treated as well: we shall consider *partial, probabilistic and composite computations* in recursive equations and in recursive program schemes. In order to obtain nondeterministic computations, we have added the powerset functor in the codomain as for the function $\mathsf{next}^+ : A \to \mathcal{P}A$. This functor extends to a *monad*, and similarly we use different monads to model different effects, an idea which goes back to Moggi [Mog91]. In addition, *distributive laws* of functors over these monads are a basic ingredient: they model the extension of effectful computations to results of other effectful computations (cf. our above extension of next^+ to sets). In conclusion, we provide a framework which is parametric in certain monads and distributive laws, but we

also state results which hold for combinations of particular monads and distributive laws.

Overview of Part II

The introduction to this part placed emphasis on the computational effect "nondeterminism"—and so do the following chapters since it seems to be the most important one and the only one for which there exists some work [Bou80, AN80, Poi82] in the classical recursive program scheme setting. However, all other effects mentioned above are examined, too.

We give a brief overview of this part which consists of Chapters 6 to 8. In Chapter 6 we see how the effects under consideration are formalized using monads, how effectful computations are formalized and how algebras with effects are composed. For the latter, we shall utilize distributive laws. This serves as a foundation for Chapter 7 which is concerned with two kinds of recursive equations with effects which are solved in algebras with effects. Moreover, the concepts from Chapter 6 are the basis for Chapter 8 where we investigate uninterpreted solutions of recursive program schemes with effects.

Throughout these chapters we mainly work in the category Set of sets and functions and related categories; only a few general definitions are given for an arbitrary category.

Chapter 6

Several Effects and Distributive Laws

The present chapter contains the concepts which are important for both the recursive equations with effects in Chapter 7 and the recursive program schemes with effects in Chapter 8. In Section 6.1 we see how effects and effectful computations are modeled using monads. We introduce five concrete monads which correspond to five effects we shall investigate throughout Part II of this thesis. We also recall that effectful computations form a Kleisli category. In Section 6.2 we then see how distributive laws can be utilized to compose algebras with effects. We prove the existence and give concrete examples of such distributive laws for all five effects.

6.1 Monads for Effectful Computations

Our aim is to allow computations with effects like termination, nondeterminism, probabilistic or composed results in connection with recursive specifications. An evident idea is to allow the algebras in which recursive equations or recursive program schemes are solved to include operations having such effects. This then affects the solutions of the recursive specifications: they must be capable of expressing the same effects. And if effects appear in solutions, there should be no drawback using them in the recursive specifications, too.

In our category theoretic view, algebras, systems of recursive equations and their solutions all are morphisms in a category $\mathcal{C}$ (cf. Definitions 2.16(1), 2.30 and 2.45); recursive program schemes and their (uninterpreted) solutions are natural transformations (cf. Definition 2.52), i. e. morphisms in the functor category $[\mathcal{C}, \mathcal{C}]$. We exploit the idea of Moggi [Mog91] and use mon-

ads in the codomains of morphisms in order to model effectful computations: the type of the effect is represented by a particular monad M, and effectful computations are given by morphisms $X \to MY$.

Next we present some concrete monads on Set and briefly explain to which types of effects they give rise. These are the effects we shall investigate in recursive equations and recursive program schemes throughout Part II of this thesis.

Examples 6.1. In the following examples X and Y are arbitrary sets and $f : X \to Y$ is an arbitrary function. We present the monads in the order in which we deal with them most often.

1. The *maybe monad* $(\mathrm{Id} + 1, \eta, \mu)$ is given as follows:

   - $(\mathrm{Id} + 1)(X) = X + 1$ (where $1 = \{\bot\}$);
   - $(\mathrm{Id} + 1)(f) = f + \mathrm{id}_1 : X + 1 \to Y + 1$;
   - $\eta_X = \mathsf{inl}_X : X \to X + 1$;
   - $\mu_X = \mathrm{id}_X + \nabla_1 : X+1+1 \to X+1$ (where $\nabla_1 = [\mathrm{id}_1, \mathrm{id}_1] : 1+1 \to 1$ is the codiagonal).

   Morphisms $X \to Y+1$ are partial functions in the sense that they map $x \in X$ to $\bot$ precisely if the partial function is undefined on x.

2. The *powerset monad* $(\mathcal{P}, \eta, \mu)$ is given as follows:

   - $\mathcal{P}X$ is the powerset of X;
   - $\mathcal{P}f : \mathcal{P}X \to \mathcal{P}Y$ is defined by $(\mathcal{P}f)(X') = f[X']$, $X' \in \mathcal{P}X$;
   - $\eta : \mathrm{Id} \to \mathcal{P}$ is given by $\eta_X(x) = \{x\}$, $x \in X$;
   - $\mu : \mathcal{P}\mathcal{P} \to \mathcal{P}$ is given by $\mu_X(S) = \bigcup_{S' \in S} S'$, $S \in \mathcal{P}\mathcal{P}X$.

   Morphisms $X \to \mathcal{P}Y$ are nondeterministic functions in the sense that they map $x \in X$ to the set of possible choices.

3. The *subdistribution monad* $(\mathcal{D}, \eta, \mu)$ is given as follows:

   - $\mathcal{D}X$ is the set of functions $d : X \to [0, 1]$ with $\sum_{x \in X} d(x) \leq 1$ (the subdistributions on X);
   - $\mathcal{D}f : \mathcal{D}X \to \mathcal{D}Y$ is defined by $(\mathcal{D}f)(d)(y) = \sum_{x \in f^{-1}(y)} d(x)$, $d \in \mathcal{D}X$, $y \in Y$ (where the sum is 0 if $f^{-1}(y) = \emptyset$);
   - $\eta : \mathrm{Id} \to \mathcal{D}$ is given by $\eta_X(x) = d$ with $d(x) = 1$ and $d(x') = 0$ for $x' \neq x$, $x, x' \in X$, i. e. by forming Dirac distributions;

- $\mu : \mathcal{DD} \to \mathcal{D}$ is given by $\mu_X(d) = e$ with $e(x) = \sum_{d' \in \mathcal{D}X} d(d') \cdot d'(x)$, $d \in \mathcal{DD}X$, $x \in X$, performing the flattening of nested probability distributions.

Morphisms $X \to \mathcal{D}Y$ are probabilistic functions: they map $x \in X$ to a (sub-)probability distribution on Y.

4. The *nonempty powerset monad* $(\mathcal{P}^+, \eta^+, \mu^+)$ is similar to the powerset monad from item (2) but leaves out the empty set in powersets:

   - $\mathcal{P}^+X$ is the set of all nonempty subsets of X;
   - $\mathcal{P}^+f : \mathcal{P}^+X \to \mathcal{P}^+Y$ is defined by $(\mathcal{P}^+f)(X') = f[X']$, $X' \in \mathcal{P}^+X$;
   - $\eta^+ : \mathrm{Id} \to \mathcal{P}^+$ is given by $\eta^+_X(x) = \{x\}$, $x \in X$;
   - $\mu^+ : \mathcal{P}^+\mathcal{P}^+ \to \mathcal{P}^+$ is given by $\mu^+_X(S) = \bigcup_{S' \in S} S'$, $S \in \mathcal{P}^+\mathcal{P}^+X$.

   Morphisms $X \to \mathcal{P}^+Y$ are proper nondeterministic functions where "proper" refers to the fact the set of possible choices must be nonempty.

5. The *environment monad* $((-)^E, \eta, \mu)$ (where E is a fixed set) is given as follows:

   - X^E is the set of functions $E \to X$, i. e. an element of X^E is an E-indexed family of elements of X;
   - $f^E : X^E \to Y^E$ is defined by $f^E((x_e)_{e \in E}) = (f(x_e))_{e \in E}$, $x_e \in X$ for all $e \in E$;
   - $\eta : \mathrm{Id} \to (-)^E$ is given by $\eta_X(x) = c_x$, $x \in X$, the constant function;
   - $\mu : ((-)^E)^E \to (-)^E$ is given by $\mu_X(((x_{e,e'})_{e' \in E})_{e \in E}) = (x_{e,e})_{e \in E}$, $x_{e,e'} \in X$ for all $e, e' \in E$ (i. e., the diagonal of an $(E \times E)$-matrix is taken).

   Morphisms $X \to Y^E$ are composite functions: they equivalently are families of functions $X \to Y$ indexed by E.

Remarks 6.2. 1. Notice that the *identity monad* $(\mathrm{Id}, \mathrm{id}, \mathrm{id})$ is the special case of the environment monad where $E = 1$. Morphisms $X \to \mathrm{Id}(Y)$ simply are (plain) functions $X \to Y$, so the effect associated with the identity monad is the "empty effect" or, to put it differently, there is no effect.

2. The first three monads M of Example 6.1 have a property in common which makes it possible to treat them in large parts of our thesis in a common framework: the sets MX carry the structure of a complete partial order with a least element (CPO, for short), see Example 7.18(1) below. This also explains why we chose the subdistribution monad rather than the more usual distribution monad where the probabilities of a distribution need to sum up to 1: we obtain CPO structures on the sets $\mathcal{D}X$ as explained in Example 7.18(1) below.

3. The reader may inquire about the choice of the five monads. There are two reasons for our choice that will become clearer as Part II of our thesis develops: first, all five monads are so-called "commutative monads" and thus admit canonical distributive laws we shall need, see Section 6.2 below. And second, either free H-algebras or free completely iterative algebras for H give rise to final or at least weakly final coalgebras for certain functors related to H, see Section 8.3. For the first three monads, this second fact is due to the CPOs mentioned in item (2). For the other two monads, it is due to the fact that there exist decomposition procedures (determinization for $\mathcal{P}^+$ and projection for $(-)^E$, which amounts to "doing nothing" in the special case of Id) which make it possible to eliminate the monad from certain diagrams and use results for the diagrams without effects to assemble results for the original diagrams.

4. As the reader may notice from the lengthy and abstract explanations in item (3), although all five monads have similar properties, there are technical differences in handling them. Indeed, in Part II of our thesis we build up a common framework for the monads, but also prove several results on the level of the particular monads.

For every monad M there is a category whose morphisms are precisely the effectful computations $X \to MY$:

Definition 6.3 (see [Kle65])**.** The *Kleisli category* $\mathcal{C}_M$ of a monad (M, η, μ) on a category $\mathcal{C}$ is given as follows:

- the objects of $\mathcal{C}_M$ are the same objects as the ones of $\mathcal{C}$;
- the morphisms of $\mathcal{C}_M$ between X and Y are all morphisms $X \to MY$ from $\mathcal{C}$;
- the identity morphism on X is $\eta_X : X \to MX$;

- composition of $f : X \to MY$ and $g : Y \to MZ$ is given by

$$X \xrightarrow{f} MY \xrightarrow{Mg} MMZ \xrightarrow{\mu_Z} MZ\ .$$

We set up some notation in connection with Kleisli categories.

Notation 6.4. When convenient, we shall draw arrows in Kleisli categories. To indicate this, we place a filled circle on the arrow stem center, i. e. an arrow $X \xrightarrow{f} MY$ in $\mathcal{C}$ is drawn as $X \xrightarrow{f}\!\!\bullet\!\!\to Y$ in $\mathcal{C}_M$.

Furthermore, for every monad M on $\mathcal{C}$ there is a canonical adjunction

$$\begin{array}{c} \mathcal{C}_M \\ {\scriptstyle J}\uparrow \dashv \downarrow{\scriptstyle V} \\ \mathcal{C} \end{array}$$

between the base category $\mathcal{C}$ and the Kleisli category $\mathcal{C}_M$. The left adjoint $J : \mathcal{C} \to \mathcal{C}_M$ is the inclusion functor given by the identity on objects and by $Jf = \eta_Y \cdot f : X \to MY$ on morphisms $f : X \to Y$. The right adjoint $V : \mathcal{C}_M \to \mathcal{C}$ is given by MX on objects X and by $Vf = \mu_Y \cdot Mf : MX \to MY$ on morphisms $f : X \to MY$. The unit of the adjunction is the monad unit η, and the counit ε is given componentwise by the identity maps id_{MX} in $\mathcal{C}$ considered as morphisms ε_X in $\mathcal{C}_M$.

Definition 6.5. A *lifting* of an endofunctor H on $\mathcal{C}$ to $\mathcal{C}_M$ is an endofunctor $\bar{H}$ on $\mathcal{C}_M$ such that $\bar{H}J = JH$.

Proposition 6.6 ([Mul94])**.** *For any functor H on $\mathcal{C}$ and monad M on $\mathcal{C}$ the following are equivalent:*

1. *there is a distributive law $\lambda : HM \to MH$ of the functor over the monad;*
2. *H lifts to $\mathcal{C}_M$.*

Remark 6.7. For a given distributive law $\lambda : HM \to MH$, the corresponding lifting $\bar{H}$ to $\mathcal{C}_M$ is given by $\bar{H}X = HX$ on objects X of $\mathcal{C}_M$ and by $\bar{H}f = \lambda_Y \cdot Hf : HX \to MHY$ on morphisms $f : X \to MY$ of $\mathcal{C}_M$.

6.2 Distributive Laws as Policies for Effect-Handling

If we try to work with effectful computations in recursive specifications, we encounter the following problem: solutions of recursive specifications are

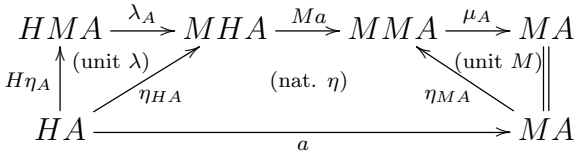
always defined as morphisms which can be decomposed into (1) the recursive specification morphism and (2) applications of the solution morphism and the algebra, see e. g. the Definitions 2.30 and 2.52 of CIAs and RPS's. Unfolding this definition which is recursive in step (2), recursive specification morphisms or algebras are applied to the results of recursive specification morphisms or algebras again. But how can this work in case the specification morphisms and algebras are effectful computations? For example, it is not possible to apply an effectful H-algebra $a : HA \to MA$ directly to HMA (where the inner MA results from previous effectful computations).

Our solution to this problem is to use a distributive law $\lambda : HM \to MH$ of the (algebra) functor H over the monad (M, η, μ) (the type of effect), see below Definition 2.15. It allows us to apply the effectful algebra $a : HA \to MA$ to HMA as follows:

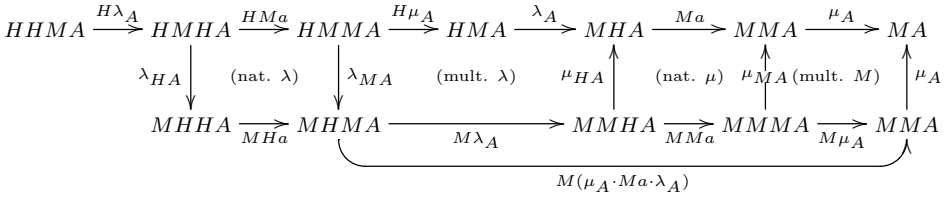
$$HMA \xrightarrow{\lambda_A} MHA \xrightarrow{Ma} MMA \xrightarrow{\mu_A} MA \tag{6.1}$$

Observe that

- this construct is an H-algebra with carrier MA;
- in case of no previous effectful computations (i. e. precomposition with $H\eta_A$) this is a "normal" application of the algebra $a : HA \to MA$ due to the unit law for λ, as the following commutative diagram shows:



- the sequential composition of such constructs can be reformulated into a nested application of the construction due to the multiplication law for λ, as the following commutative diagram shows:



- due to naturality of λ this approach does not depend on the algebra carrier but only on the algebra functor: this is purely syntactic and thus can also be used in case of uninterpreted solutions of recursive program schemes as we will see in Chapter 8.

To summarize, distributive laws of functors over monads can be viewed as "policies for effect-handling".

The next questions are, of course: do there exist distributive laws of algebra functors over the monads from Example 6.1? And are they "reasonable" policies for effect-handling? Moreover, do we need to find a new distributive law for every combination of algebra functor and monad? We shall answer these questions in the remainder of this section.

Canonical Distributive Laws for Polynomial Functors

In Remark 6.2(3) we stated that our monads of interest are all *commutative*. We shall explain this next.

Definition 6.8 (see [Mac98])**.** A *monoidal category* $(\mathcal{C}, I, \otimes)$ consists of a category $\mathcal{C}$, a bifunctor $\otimes : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$ and an object $I \in \mathcal{C}$ as well as three natural isomorphisms

$$\begin{aligned} \mathsf{ul}_X &: I \otimes X \to X \\ \mathsf{ur}_X &: X \otimes I \to X \\ \mathsf{a}_{X,Y,Z} &: (X \otimes Y) \otimes Z \to X \otimes (Y \otimes Z) \end{aligned}$$

such that the following diagrams commute:

$$\begin{array}{ccc} (X \otimes I) \otimes Y & \xrightarrow{\mathsf{a}_{X,I,Y}} & X \otimes (I \otimes Y) \\ & {\scriptstyle \mathsf{ur}_X \otimes \mathrm{id}_Y} \searrow \quad \swarrow {\scriptstyle \mathrm{id}_X \otimes \mathsf{ul}_Y} & \\ & X \otimes Y & \end{array}$$

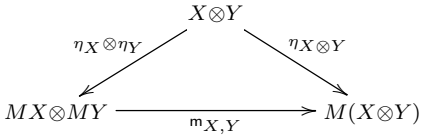
$$\begin{array}{ccccc} ((W \otimes X) \otimes Y) \otimes Z & \xrightarrow{\mathsf{a}_{W,X,Y} \otimes \mathrm{id}_Z} & (W \otimes (X \otimes Y)) \otimes Z & \xrightarrow{\mathsf{a}_{W,X \otimes Y,Z}} & W \otimes ((X \otimes Y) \otimes Z) \\ {\scriptstyle \mathsf{a}_{W \otimes X,Y,Z}} \downarrow & & & & \downarrow {\scriptstyle \mathrm{id}_W \otimes \mathsf{a}_{X,Y,Z}} \\ (W \otimes X) \otimes (Y \otimes Z) & & \xrightarrow[\mathsf{a}_{W,X,Y \otimes Z}]{} & & W \otimes (X \otimes (Y \otimes Z)) \,. \end{array}$$

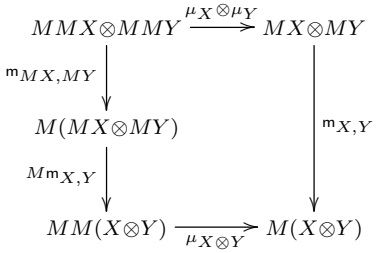
If there is furthermore a natural isomorphism $\mathsf{c}_{X,Y} : X \otimes Y \to Y \otimes X$ which makes the diagrams

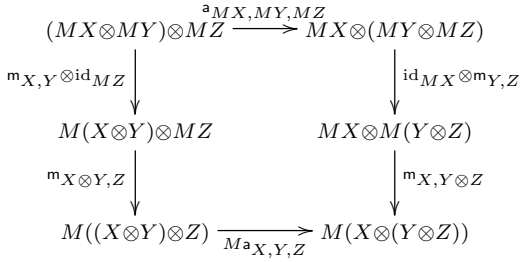
$$\begin{array}{ccc} (X \otimes Y) \otimes Z & \xrightarrow{\mathsf{a}_{X,Y,Z}} & X \otimes (Y \otimes Z) \\ {\scriptstyle \mathsf{c}_{X,Y} \otimes \mathrm{id}_Z} \downarrow & & \downarrow {\scriptstyle \mathsf{c}_{X,Y \otimes Z}} \\ (Y \otimes X) \otimes Z & & (Y \otimes Z) \otimes X \\ {\scriptstyle \mathsf{a}_{Y,X,Z}} \downarrow & & \downarrow {\scriptstyle \mathsf{a}_{Y,Z,X}} \\ Y \otimes (X \otimes Z) & \xrightarrow[\mathrm{id}_Y \otimes \mathsf{c}_{X,Z}]{} & Y \otimes (Z \otimes X) \end{array} \qquad \begin{array}{ccc} X \otimes (Y \otimes Z) & \xrightarrow{\mathsf{a}^{-1}_{X,Y,Z}} & (X \otimes Y) \otimes Z \\ {\scriptstyle \mathrm{id}_X \otimes \mathsf{c}_{Y,Z}} \downarrow & & \downarrow {\scriptstyle \mathsf{c}_{X \otimes Y,Z}} \\ X \otimes (Z \otimes Y) & & Z \otimes (X \otimes Y) \\ {\scriptstyle \mathsf{a}^{-1}_{X,Z,Y}} \downarrow & & \downarrow {\scriptstyle \mathsf{a}^{-1}_{Z,X,Y}} \\ (X \otimes Z) \otimes Y & \xrightarrow[\mathsf{c}_{X,Z} \otimes \mathrm{id}_Y]{} & (Z \otimes X) \otimes Y \end{array}$$

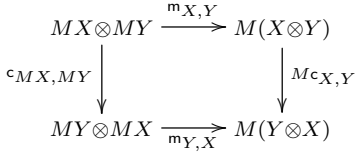
commute and for which we have $\mathsf{c}_{Y,X} \cdot \mathsf{c}_{X,Y} = \mathrm{id}_{X \otimes Y}$ for all objects X, Y, it is called *symmetric monoidal*.

Definition 6.9. A *symmetric monoidal monad* on a symmetric monoidal category $(\mathcal{C}, I, \otimes, \mathsf{ul}, \mathsf{ur}, \mathsf{a}, \mathsf{c})$ is a monad (M, η, μ) together with a transformation $\mathsf{m}_{X,Y} : MX \times MY \to M(X \times Y)$ natural in X and Y (called *double strength*) such that the following diagrams commute:









To simplify the law for a we just write $\mathsf{m}_{X \otimes Y,Z} \cdot (\mathsf{m}_{X,Y} \otimes \mathrm{id}_{MZ}) = \mathsf{m}_{X,Y \otimes Z} \cdot (\mathrm{id}_{MX} \otimes \mathsf{m}_{Y,Z})$.

We do not present the formal definition of a commutative monad since we shall not need it. Instead, we work with the following characterization by Kock:

Theorem 6.10 ([Koc70, Koc72])**.** *A monad is commutative if and only if it has the structure of a symmetric monoidal monad.*

In the remainder of this thesis, we shall use the term "commutative monad" rather than "symmetric monoidal monad". We are only interested in commutative monads on Set considered as a symmetric monoidal category $(\mathsf{Set}, 1, \times, \mathsf{ul}, \mathsf{ur}, \mathsf{a}, \mathsf{c})$ with the singleton set 1 and the cartesian product $\times$.

Examples 6.11. All five monads from Example 6.1 are commutative monads with a (unique) double strength $\mathsf{m}_{X,Y} : MX \times MY \to M(X \times Y)$ (cf. [HJS07] for the first three examples):

1. For the maybe monad we have $\mathsf{m}_{X,Y}(x, y) = \begin{cases} (x, y) & x \in X, y \in Y \\ \bot & x = \bot \text{ or } y = \bot \end{cases}$;

2. for the powerset monad $\mathsf{m}_{X,Y}(S,T) = S \times T$ is the cartesian product;

3. for the subdistribution monad we have $\mathsf{m}_{X,Y}(d,e) = f$ where $f(x,y) = d(x) \cdot e(y)$;

4. for the nonempty powerset monad $\mathsf{m}_{X,Y}(S,T) = S \times T$ is similar to the powerset monad; and

5. for the environment monad $\mathsf{m}_{X,Y}((x_e)_{e\in E},(y_e)_{e\in E}) = ((x_e,y_e))_{e\in E}$ is an isomorphism (for the special case $E = 1$ of the identity monad this simply is $\mathsf{m}_{X,Y} = \mathrm{id}_{X\times Y}$).

Lemma 6.12 ([HJS07])**.** *There exist canonical distributive laws $\lambda : HM \to MH$ of every finitary polynomial endofunctor H on* Set *over every commutative monad M on* Set.

Remark 6.13. The proof of Lemma 6.12 is given in [HJS07] by induction on the structure of a finitary polynomial set functor H. In fact, any such functor is built from constant functors and the identity functor using finite products and possibly infinite coproducts. We recall the respective constructions of the canonical distributive laws of such H over any commutative monad M:

1. Let $H = C_A$ be the constant functor with value A. A distributive law $\lambda : C_A M \to MC_A$ of H over M is given by $\lambda_X = \eta_A$ for every object X.

2. Let $H = \mathrm{Id}$ be the identity functor. Clearly $\lambda = \mathrm{id} : \mathrm{Id}M \to M\mathrm{Id}$ is a distributive law of H over M.

3. Let $H = H_1 \times H_2$ where H_1 and H_2 are functors with distributive laws $\lambda_1 : H_1M \to MH_1$ and $\lambda_2 : H_2M \to MH_2$. Then we have a distributive law λ of $H_1 \times H_2$ over M given by

$$(H_1 \times H_2)M = H_1M \times H_2M \xrightarrow{\lambda_1\times\lambda_2} MH_1 \times MH_2 \xrightarrow{\mathsf{m}} M(H_1 \times H_2)\,.$$

   Here we abuse notation and write m for the natural transformation with components $\mathsf{m}_{H_1X,H_2X} : MH_1X \times MH_2X \to M(H_1X \times H_2X)$ arising from the double strength m of the commutative monad M, see Theorem 6.10 and Definition 6.9.

4. Let $H = \coprod_{i\in I} H_i$ where each endofunctor H_i has a distributive law $\lambda_i : H_iM \to MH_i$ over the monad M. Then we have a distributive law of H over M as follows:

$$(\textstyle\coprod_{i\in I} H_i)M = \coprod_{i\in I} H_iM \xrightarrow{\coprod_{i\in I}\lambda_i} \coprod_{i\in I} MH_i \xrightarrow{[M\mathsf{inj}_i]} M(\coprod_{i\in I} H_i)$$

where $\mathsf{inj}_i : H_i \to \coprod_{i \in I} H_i$ denote the coproduct injections and $[M\mathsf{inj}_i]$ is the unique natural transformation with $[M\mathsf{inj}_i] \cdot \mathsf{inj}_j = M\mathsf{inj}_j$ for all $j \in I$.

Examples 6.14. Let H be a finitary polynomial set functor associated to a signature Σ. We make the distributive law λ of Lemma 6.12 explicit for our five monads from Example 6.1 and verify that they are "reasonable" policies for effect handling.

1. For the maybe monad $M = \mathrm{Id} + 1$, $\lambda_X : H(X + 1) \to HX + 1$ is given by

$$\lambda_X(\sigma(x_1, \ldots, x_n)) = \begin{cases} \sigma(x_1, \ldots, x_n) & x_i \in X \text{ for all } 1 \leq i \leq n \\ \bot & \text{else} \end{cases}$$

   for every n-ary operation symbol $\sigma \in \Sigma_n$. According to (6.1), a partial algebra (for a finitary polynomial set functor) is strictly extended to possibly undefined ($\bot$) arguments, i. e. it yields the undefined result $\bot$ whenever at least one argument is undefined. This matches the idea of composing partial computations.

2. For the powerset monad $M = \mathcal{P}$, $\lambda_X : H\mathcal{P}X \to \mathcal{P}HX$ acts as follows:

$$\lambda_X(\sigma(X_1, \ldots, X_n)) = \{\sigma(x_1, \ldots, x_n) \mid x_i \in X_i, 1 \leq i \leq n\}$$

   for every $\sigma \in \Sigma_n$ and $X_i \in \mathcal{P}X$, $1 \leq i \leq n$. According to (6.1), a nondeterministic algebra is applied to arguments which are sets of possible choices by applying it to each combination of possible choices from the arguments, resulting in a set of all possible results. This matches the idea of composing nondeterministic computations.

3. For $M = \mathcal{D}$, $\lambda_X : H\mathcal{D}X \to \mathcal{D}HX$ is given by

$$\lambda_X(\sigma(d_1, \ldots, d_n)) = f$$

   for every $\sigma \in \Sigma_n$ and $d_i \in \mathcal{D}X$, $1 \leq i \leq n$, where

$$f(\tau(x_1, \ldots, x_m)) = \begin{cases} d_1(x_1) \cdot \cdots \cdot d_n(x_m) & \tau = \sigma \\ 0 & \tau \neq \sigma\,. \end{cases}$$

   According to (6.1), a probabilistic algebra is applied to arguments which are subdistributions on A by applying it to each combination of elements from A and computing the overall probabilities for every result from A. This matches the idea of composing probabilistic computations.

4. For $M = \mathcal{P}^+$, $\lambda_X : H\mathcal{P}^+X \to \mathcal{P}^+HX$ acts similar as for $\mathcal{P}$:

$$\lambda_X(\sigma(X_1, \ldots, X_n)) = \{\sigma(x_1, \ldots, x_n) \mid x_i \in X_i, 1 \leq i \leq n\}$$

for every $\sigma \in \Sigma_n$ and $X_i \in \mathcal{P}^+X$, $1 \leq i \leq n$. The only difference is that all arguments and results are nonempty.

5. For the environment monad $M = (-)^E$, the distributive law $\lambda_X : HX^E \to (HX)^E$ is given by

$$\lambda_X(\sigma((x_{1,e})_{e\in E}, \ldots, (x_{n,e})_{e\in E})) = (\sigma(x_{1,e}, \ldots, x_{n,e}))_{e\in E}$$

for every $\sigma \in \Sigma_n$. According to (6.1), an E-composite algebra is applied to E-composite arguments by applying its e-component to the e-components of the arguments for every $e \in E$. This matches the idea of composing composite computations.
For the special case $E = 1$ of the identity monad $M = \mathrm{Id}$ we obtain from Lemma 6.12 the identity transformation $\lambda = \mathrm{id} : H \to H$. Then the extended algebra (6.1) is just the original plain algebra. This matches the fact that plain algebras can be composed without using distributive laws.

Canonical Distributive Laws for Analytic Functors

Next we shall generalize Lemma 6.12 from finitary polynomial set functors to analytic functors.

Notation 6.15. We denote by NatB the category of natural numbers and bijections, and by $\mathcal{E} : \mathsf{NatB} \to \mathsf{Set}$ the (non-full) embedding.

Definition 6.16 ([Joy81, Joy86])**.** An endofunctor H on Set is called *analytic* provided that it is the left Kan extension of a functor $\mathsf{NatB} \to \mathsf{Set}$ along $\mathcal{E}$.

Remarks 6.17. 1. In fact, Joyal defined analytic functors by explicitly stating what these Kan extensions are. Let $\mathcal{S}_n$ be the symmetric group of all permutations of n. For every subgroup G of $\mathcal{S}_n$ the *symmetrized representable functor* sends each set X to the set X^n/G of orbits under the action of G on X^n by coordinate interchange, i. e., X^n/G is the quotient of X^n modulo the equivalence $\sim_G$ with $(x_1, \ldots, x_n) \sim_G (y_1, \ldots, y_n)$ iff $(x_{p(1)}, \ldots, x_{p(n)}) = (y_1, \ldots, y_n)$ for some $p \in G$. It is straightforward to work out that an endofunctor on Set is analytic iff it is a coproduct of symmetrized representables. So every analytic functor H can be written in the form

$$HX = \coprod_{n\in\mathbb{N},\, G\leq\mathcal{S}_n} A_{n,G} \times X^n/G\,. \tag{6.2}$$

2. Notice that by (6.2) an analytic functor is a quotient of the corresponding polynomial functor P with $PX = \coprod_{n,G} A_{n,G} \times X^n$.

3. Clearly every analytic functor is finitary. Joyal proved in [Joy81, Joy86] that a finitary endofunctor on Set is analytic iff it weakly preserves wide pullbacks.

Examples 6.18. 1. Let Σ be a finitary signature of operation symbols. Clearly the associated polynomial functor H_Σ is analytic (take $A_{n,G} = \Sigma_n$ for the trivial subgroup $G = \{\mathrm{id}\} \leq \mathcal{S}_n$ and $A_{n,G} = 0$ else).

2. The functor H assigning to a set X the set of finite multisets over X is analytic since it has the form $HX = \coprod_{n\in\mathbb{N}} X^n/\mathcal{S}_n$.

3. The functor H assigning to a set X the set of (rooted and ordered) trees with nodes labeled in X is analytic. In fact, H is the left Kan extension of $t : \mathsf{NatB} \to \mathsf{Set}$ assigning to the natural number n the set $t(n)$ of trees with $\{0, \ldots, n-1\}$ as the set of nodes.

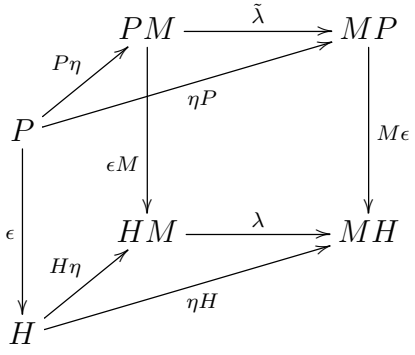
4. The finite powerset functor $\mathcal{P}_\mathrm{f}$ is not analytic as it does not preserve weak wide pullbacks.

Theorem 6.19. *Let H be an analytic functor and let M be a commutative monad on Set. Then there exists a canonical distributive law λ of H over M.*

For the proof of Theorem 6.19 we need the following

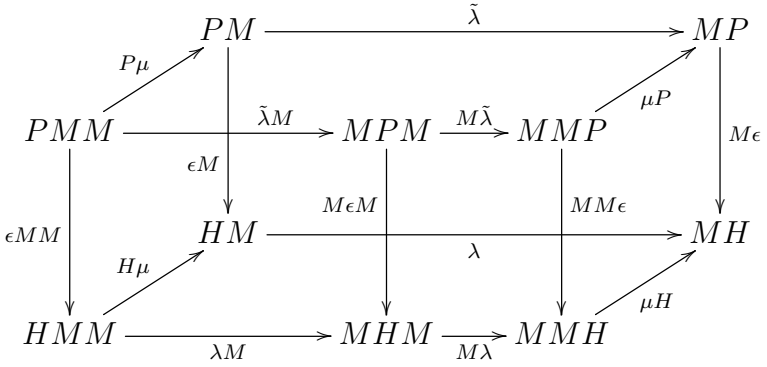
Lemma 6.20. *Let H be a quotient of the functor P via $\epsilon : P \to H$. Let $\tilde{\lambda} : PM \to MP$ be a distributive law of P over the monad M and let $\lambda : HM \to MH$ be a natural transformation such that $M\epsilon \cdot \tilde{\lambda} = \lambda \cdot \epsilon M$. Then λ is a distributive law of H over the monad M.*

Proof. We verify the unit and multiplication laws for λ. For the unit law consider the diagram below:



The upper triangle commutes by the unit law for $\tilde{\lambda}$. The squares commute by naturality of η, by naturality of ϵ and by the assumption $M\epsilon \cdot \tilde{\lambda} = \lambda \cdot \epsilon M$. Hence, the desired lower triangle commutes when precomposed with ϵ. Thus, this triangle commutes since ϵ is epimorphic.

For the multiplication law consider the following diagram



Here the upper square commutes by the multiplication law for $\tilde{\lambda}$. The squares forming the cuboid sides commute due to naturality of μ (right-hand side), to naturality of ϵ (left-hand side) and due to the assumption $M\epsilon \cdot \tilde{\lambda} = \lambda \cdot \epsilon M$ (the other sides). This proves that the desired lower square commutes when precomposed with the epimorphism ϵMM, whence this square commutes. □

Proof of Theorem 6.19. We construct a canonical distributive law of any symmetrized representable endofunctor over any commutative monad M. Then by using Remark 6.13, items (1), (3) and (4) we obtain a distributive law for every endofunctor of the form (6.2) over M, which is the desired result.

Let H be the symmetrized representable functor $HX = X^n/G$ for some natural number n and some subgroup G of $\mathcal{S}_n$. Recall that each permutation $p \in G$ is a composite of transpositions $(i\ i+1)$ of neighboring elements. Let γ_{pX} denote the bijection $X^n \to X^n$ corresponding to p. For $p = (i\ i+1)$ this is the map swapping the i-th and $(i+1)$-st components of the product, and we simply write γ_{iX}. For the endofunctor $Q_nX = X^n$ on Set we see that $\gamma_p, \gamma_i : Q_n \to Q_n$ are natural transformations. Moreover, H is a quotient of Q_n via $\epsilon : Q_n \to H$, where each component ϵ_X is the coequalizer of all the automorphisms γ_{pX}, $p \in G$, on X^n.

We have distributive laws $\lambda_n : Q_nM \to MQ_n$ of the functors Q_n over the monad M defined by induction on n as follows: $\lambda_1 = \mathrm{id} : \mathrm{Id}M \to M\mathrm{Id}$ is the trivial distributive law from Remark 6.13(2) and

$$\lambda_{n+1} = (Q_{n+1}M = Q_nM \times M \xrightarrow{\lambda_n \times \mathrm{id}} MQ_n \times M \xrightarrow{\mathsf{m}} M(Q_n \times \mathrm{Id}) = MQ_{n+1})$$

is a distributive law by Remark 6.13(3) above.

We prove that for any n and any γ_p the following square commutes:

$$\begin{array}{ccc} Q_nM & \xrightarrow{\lambda_n} & MQ_n \\ {\scriptstyle \gamma_p M}\downarrow & & \downarrow{\scriptstyle M\gamma_p} \\ Q_nM & \xrightarrow{\lambda_n} & MQ_n\,. \end{array} \tag{6.3}$$

In fact, it is sufficient to prove this for each γ_i, $1 \leq i < n$, because every γ_p is a composite of maps of the form γ_i. Notice that each component of γ_i is of the form $\mathrm{id}^{i-1} \times \mathsf{c} \times \mathrm{id}^{n-i-1}$. We prove that diagram (6.3) commutes for each γ_i by induction on n.

Base cases: for $n = 1$ there is nothing to prove; for $n = 2$ there are two cases: the case $\gamma_i = \mathrm{id}$ is obvious, and for $\gamma_{iX} = \mathsf{c}_{X,X}$ on $X \times X$ we simply use that M is symmetric monoidal (cf. Definition 6.9): we have $\lambda_2 = \mathsf{m} : M(-) \times M(-) \to M(- \times -)$ and so the desired square in (6.3) simply is the commutative diagram

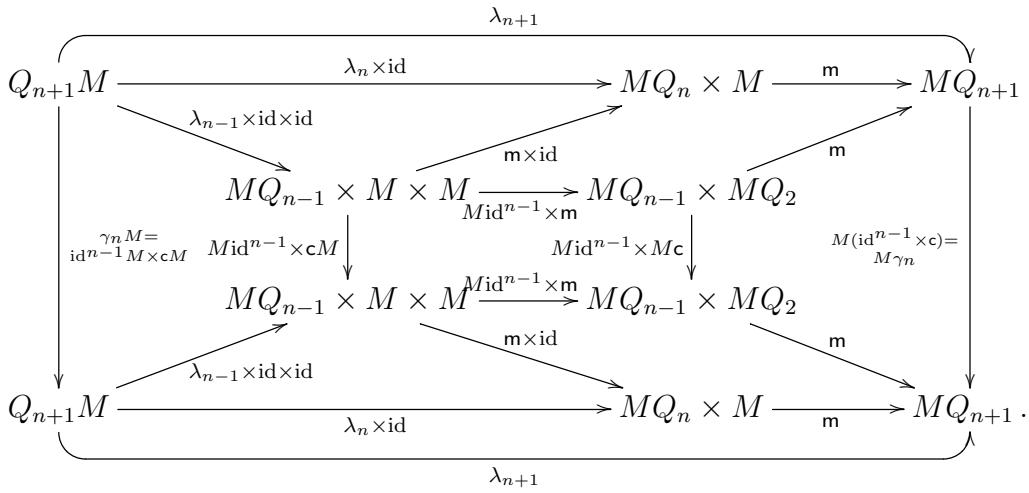
$$\begin{array}{ccc} Q_2M & \xrightarrow{\mathsf{m}} & MQ_2 \\ {\scriptstyle \mathsf{c}M}\downarrow & & \downarrow{\scriptstyle M\mathsf{c}} \\ Q_2M & \xrightarrow{\mathsf{m}} & MQ_2\,. \end{array}$$

Induction step (from n to $n+1$, $n \geq 2$): here we distinguish two cases. For the case $i < n$ consider the diagram

$$\begin{array}{ccccccc} & & & & \overset{\lambda_{n+1}}{\frown} & & \\ Q_{n+1}M & = & Q_nM \times M & \xrightarrow{\lambda_n \times \mathrm{id}} & MQ_n \times M & \xrightarrow{\mathsf{m}} & M(Q_n \times \mathrm{Id}) = MQ_{n+1} \\ {\scriptstyle \gamma_i M}\downarrow & & {\scriptstyle \gamma_i M \times \mathrm{id} M}\downarrow & & {\scriptstyle M\gamma_i \times M\mathrm{id}}\downarrow & & \downarrow{\scriptstyle M(\gamma_i \times \mathrm{id})} \qquad \downarrow{\scriptstyle M\gamma_i} \\ Q_{n+1}M & = & Q_nM \times M & \xrightarrow[\lambda_n \times \mathrm{id}]{} & MQ_n \times M & \xrightarrow[\mathsf{m}]{} & M(Q_n \times \mathrm{Id}) = MQ_{n+1}\,. \\ & & & & \underset{\lambda_{n+1}}{\smile} & & \end{array}$$

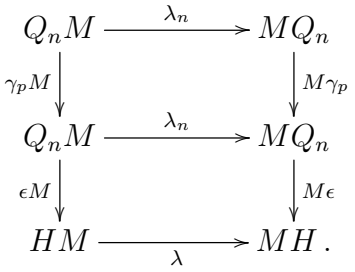
The left-hand square in the middle commutes by the induction hypothesis, the right-hand one by naturality of m and the upper and lower parts by the definition of λ_{n+1}.

For the case $i = n$ consider the diagram



The upper and lower parts as well as the two triangles commute by the definition of λ_n. The left-hand square trivially commutes, the middle one commutes due to symmetric monoidality of M and the right-hand one due to naturality of m. The remaining two squares commute since m is associative, see Definition 6.9. This completes the verification of (6.3).

To complete the proof as well consider the diagram



The upper squares are instances of the commutative squares in (6.3) for each permutation $p \in G$. Thus from the universality of the coequalizer ϵ_{MX} we obtain the unique map λ_X such that the lower square commutes. It is not difficult to see that λ is a natural transformation, whence it is a distributive law by Lemma 6.20. □

Remark 6.21. Let M be a commutative monad and let H be an analytic endofunctor. Write H as a quotient of the polynomial functor P, see Remark 6.17(2), and denote by $\epsilon : P \to H$ the natural transformation formed by the canonical surjections. It follows from the proof of Theorem 6.19 that for the canonical distributive laws $\tilde{\lambda} : PM \to MP$ and $\lambda : HM \to MH$ we have

$$M\epsilon \cdot \tilde{\lambda} = \lambda \cdot \epsilon M : PM \to MH\,. \tag{6.4}$$

Example 6.22. Let H be the finite multiset functor of Example 6.18(2). Its canonical distributive law λ over the powerset monad $M = \mathcal{P}$ is given by

$$\lambda_X(\langle X_1, \ldots, X_n\rangle) = \{\langle x_1, \ldots, x_n\rangle \mid x_i \in X_i, 1 \leq i \leq n\}$$

for $X_j \subseteq X$, $1 \leq j \leq n$, where the angular brackets denote multisets. In fact, this follows from equation (6.4) and Example 6.14(2) above, applied to the polynomial functor $PX = \coprod_{n\in\mathbb{N}} X^n$.

Further Canonical and Non-Canonical Distributive Laws

Specializing to fixed functors H or monads M, Theorem 6.19 may be extended to obtain further canonical distributive laws. This is demonstrated in the following

Examples 6.23.

1. For the powerset monad $M = \mathcal{P}$ and nonempty powerset monad $M = \mathcal{P}^+$, there exist canonical distributive laws of every endofunctor H which weakly preserves pullbacks over M. This was proved directly in [Jac04], but for $M = \mathcal{P}$ it already follows from the result that H lifts to Set_M (the category of sets and relations) iff H weakly preserves pullbacks which was proved in [Trn77] and [CKW90]. This includes e. g. the finite powerset functor $H = \mathcal{P}_\mathrm{f}$ which is not analytic as we have seen in Example 6.18(4).

2. For the environment monad $M = (-)^E$, there exists a canonical distributive law of every endofunctor H over M as follows: observe that $X^E \cong \prod_{i\in E} X$ with projections $\pi_i^X : X^E \to X$ for each $i \in E$. Define $\lambda_X : H(X^E) \to (HX)^E$ as the unique morphism such that $\pi_i^{HX} \cdot \lambda_X = H\pi_i^X$ for every $i \in E$. It is easy to prove that λ is a distributive law of H over M and that this generalizes the canonical distributive laws from Theorem 6.19. For the special case $E = 1$ of the identity monad $M = \mathrm{Id}$, the canonical distributive law of every endofunctor H over M simply is $\lambda = \mathrm{id}$.

3. For the identity functor $H = \mathrm{Id}$, distributive laws of H over M are precisely the monad endomorphisms on M. For example, $\mathrm{id} : M \to M$ clearly is a monad morphism on M for every monad M.

Distributive laws of functors over monads seem to be rather rare. For example, there are no further distributive laws of functors over the powerset/nonempty powerset monads than the canonical ones from Theorem 6.19. This follows from [Trn77, CKW90] as explained in Example 6.23(1). However, for other commutative monads further distributive laws can be found:

Examples 6.24. Let us come back to Example 6.23(3). Besides the identity monad endomorphisms we mentioned there, there may be further monad endomorphisms:

1. Consider the environment monad $M = (-)^E$ for $|E| = 2$, i. e. $MX = X \times X$. Then $\mathsf{c} : M \to M$ is a monad endomorphism where we abuse notation and write c for the natural transformation with components $\mathsf{c}_{X,X}$ arising from the symmetry isomorphism c of the symmetric monoidal category Set.

2. For any monoid (O, e, m) the *output monad* (M, η, μ) is given by $MX = X \times O$, $\eta_X(x) = (x, e)$ and $\mu_X(((x, o_1), o_2)) = (x, m(o_1, o_2))$. This is a commutative monad precisely if the monoid is commutative. Every monoid endomorphism $h : O \to O$ extends to a monad endomorphism $\mathrm{id} \times h : M \to M$, and in general there may be more than one monoid endomorphism: consider e. g. the commutative monoid $(\mathbb{N}, 0, +)$. Here for every $n \in \mathbb{N}$ multiplication with n is a monoid endomorphism since 0 is an absorbing element for multiplication and we have the distributive law of multiplication over addition.

6.3 Related Work

The idea to model computational effects with monads is due to Moggi [Mog91]. This proved to be a successful concept and was implemented in several programming languages, see e. g. Wadler's paper [Wad95]. Monads, however, have been invented much earlier, see [HP07] for a historical overview.

Another approach to effects is Lawvere theories which are related to monads [Pow06]. This approach was taken by Abou-Saleh and Pattinson [ASP11] who extend a programming language syntax by effects and accordingly extend the operational semantics. As we do for some effects in Chapter 8, they explore final coalgebras in CPO-enriched Kleisli categories for a denotational semantics. However, both their and our approach seem to work for different effects.

Kleisli categories are named after their inventor [Kle65]. Whereas other types of distributive laws have been proved equivalent to certain liftings of functors since distributive laws have been invented by Beck (cf. [Bec69] page 122), Mulry [Mul94] seems to be the first one to relate distributive laws and liftings to Kleisli categories.

Analytic functors have been defined and explored by Joyal [Joy81, Joy86, JS93]. Hasuo, Jacobs and Sokolova described how to construct canonical dis-

tributive laws of polynomial functors over commutative monads in [HJS07]; their result was extended from polynomial to analytic functors in our joint work [MPS09] with Milius and Palm.

Chapter 7

Recursive Equations and Effects

In this chapter, we introduce and investigate two formats of (systems of) recursive equations and the algebras with effects in which they have unique solutions. Whereas their definitions and first results are uniform, we need to consider the effects corresponding to the monads from Example 6.1 separately as we proceed.

We give a brief overview of the present chapter. In Section 7.1 we introduce Kleisli-CIAs and λ-CIAs and prove basic results about the categories they form. Section 7.2 is concerned with the free Kleisli-CIAs and λ-CIAs. A characterization of Kleisli-CIAs and λ-CIAs is provided in Section 7.3 for the effects given by the maybe monad $\mathrm{Id}+1$, the powerset monad $\mathcal{P}$ and the environment monad $(-)^E$.

Assumption 7.1. Throughout the rest of Part II of this thesis we assume that $\lambda : HM \to MH$ is a distributive law of a functor H over a monad (M, η, μ). The corresponding lifting of H to $\mathcal{C}_M$ (cf. Proposition 6.6) is denoted by $\bar{H}$.

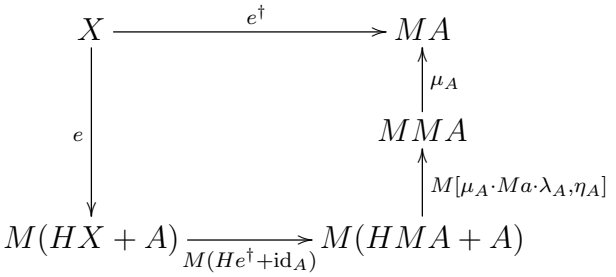
7.1 Kleisli-CIAs and λ-CIAs

We consider CIAs in the two categories Set and Set_M. To distinguish CIAs for an endofunctor H on Set from those for a lifting $\bar{H}$ we have the following

Definition 7.2. We call a CIA for a lifting $\bar{H} : \mathsf{Set}_M \to \mathsf{Set}_M$ a *Kleisli-CIA*.

Remark 7.3. If we spell out the definition of a CIA in Set_M, we see that a flat equation morphism is a map $e : X \to M(HX + A)$, and a solution of e

in the algebra $a : HA \to MA$ is a map $e^\dagger : X \to MA$ such that the diagram



commutes. This means that the algebra (A, a) as well as the recursive equation e and its solution are "effectful".

Example 7.4. Let $M = \mathcal{P}$ and let $H = H_\Sigma$ be the polynomial functor for a signature Σ with two binary operation symbols $+$ and $*$. The nondeterministic recursive equation

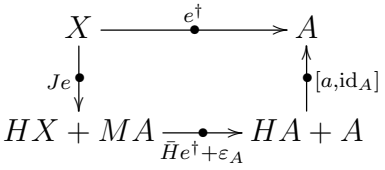
$$x = \{x + x, x * x, a_0\} \quad \text{where } x \in X \text{ and } a_0 \in A \tag{7.1}$$

gives rise to a flat equation morphism. As we shall see in Example 7.52(2), the set $A = \{2, 3, \ldots, 100\} \subset \mathbb{N}$ together with the operations $+, * : A \times A \to \mathcal{P}A$ returning the sum and product, respectively, as a singleton set if this is less than or equal to 100 and $\emptyset$ otherwise, is a Kleisli-CIA. If we choose $a_0 = 10$ in the above equation (7.1), the unique solution of x is $\{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\} \subseteq A$.

Observe that, in general, the notion of a Kleisli-CIA automatically connects effectful operations of an algebra with effectful recursive equations. One might want to consider these two separately. To this end we propose the following concept of λ-CIAs as a notion of algebras with effectful operations, where effects in recursive equations are allowed in the parameters only. As we shall see, an $\bar{H}$-algebra on A is a λ-CIA iff its related H-algebra on MA is an "ordinary" CIA.

Recall Notation 6.4 for Kleisli categories.

Definition 7.5. For the monad M on Set a (flat) *M-equation morphism* (with parameters in A) is a morphism $e : X \to HX + MA$. A solution of e in the $\bar{H}$-algebra $a : HA \nrightarrow A$ is a morphism $e^\dagger : X \nrightarrow A$ such that the diagram



(7.2)

commutes in Set_M. A *completely λ-iterative algebra* (or *λ-CIA*, for short) for H is an $\bar{H}$-algebra (A, a) in which every M-equation morphism with parameters in A has a unique solution.

Remark 7.6. Observe that an $\bar{H}$-algebra (A, a) is a λ-CIA iff $\mu_A \cdot Ma \cdot \lambda_A : HMA \to MA$ is an (ordinary) CIA for the endofunctor H on Set. In fact, this is trivial to see by writing diagram (7.2) in Set:

$$\begin{array}{ccc} X & \xrightarrow{\quad e^\dagger \quad} & MA \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [\mu_A \cdot Ma \cdot \lambda_A, \mathrm{id}_{MA}]} \\ HX + MA & \xrightarrow[He^\dagger + \mathrm{id}_{MA}]{} & HMA + MA \end{array}$$

Also notice that in e the monad M is applied only to the second component of the coproduct in the codomain, whereas in a flat equation morphism with respect to a Kleisli-CIA M is applied to the whole coproduct, cf. Remark 7.3.

Example 7.7. Continuing Example 7.4 for $M = \mathcal{P}$, we see that the formal equation in (7.1) does not give rise to an M-equation morphism with parameters in A, but the system

$$\begin{aligned} x &= y * z \\ y &= \{2, 3\} \\ z &= \{4, 5\} \end{aligned}$$

does. Its unique solution assigns to x the set $\{8, 10, 12, 15\}$. In fact, as we shall see in Proposition 7.12, A also is a λ-CIA. However, as we shall see in Example 7.13, the concepts of Kleisli-CIAs and λ-CIAs are different in general.

Notation 7.8. Let H_Σ be a finitary polynomial functor, and let $a : H_\Sigma A \to MA$ be an $\bar{H}_\Sigma$-algebra. We denote by $\sigma^A : A^n \to MA$ the component of $a : H_\Sigma A \to MA$ corresponding to the n-ary operation symbol σ of Σ.

Remark 7.9. We analyze the meaning of diagram (7.2) for a finitary polynomial functor and the distributive laws of Examples 6.14, (1)–(5). Notice that in this case an M-equation morphism e corresponds to a system of equations where the right-hand sides are of the form $\sigma(x_{i_1}, \ldots, x_{i_k})$, $\sigma \in \Sigma_k$, or are elements of MA. Further notice that we always have $e^\dagger(x) = e(x)$ if $e(x) \in MA$. We describe the meaning of a solution of an equation $x = \sigma(x_1, \ldots, x_k)$.

1. For the maybe monad $MX = X + \{\bot\}$, we have $e^\dagger(x) = \bot$ if $e^\dagger(x_i) = \bot$ for some $1 \le i \le k$, otherwise $e^\dagger(x) = \sigma^A(e^\dagger(x_1), \ldots, e^\dagger(x_k)) \in A + \{\bot\}$.

2. For the powerset monad $M = \mathcal{P}$, we have

$$e^\dagger(x) = \bigcup\{\sigma^A(a_1, \dots, a_k) \mid a_i \in e^\dagger(x_i)\}\,.$$

3. For the subdistribution monad $M = \mathcal{D}$, we have $e^\dagger(x) = d$ where

$$d(b) = \sum_{d' = \sigma^A(a_1,\dots,a_k)} e^\dagger(x_1)(a_1) \cdot \dots \cdot e^\dagger(x_k)(a_k) \cdot d'(b)\,.$$

4. For the nonempty powerset monad $M = \mathcal{P}^+$, the equation has the analogous meaning as for $M = \mathcal{P}$ in item (2), the only difference is that all sets occurring in the equation are nonempty.

5. For the environment monad $MX = X^E$, we have

$$e^\dagger(x)(i) = \sigma^A(e^\dagger(x_1)(i), \dots, e^\dagger(x_k)(i))(i)$$

for all $i \in E$.
For the special case $E = 1$ (i. e. the identity monad $M = \mathrm{Id}$), we obtain $e^\dagger(x) = \sigma^A(e^\dagger(x_1), \dots, e^\dagger(x_k))$ as for ordinary CIAs (cf. diagram (2.1)).

We give further examples of λ-CIAs:

Examples 7.10. 1. Consider the maybe monad $MX = X + \{\bot\}$ and the distributive law λ of a finitary polynomial functor H_Σ over M of Example 6.14(1). Let FY be the algebra of finite Σ-trees on Y from Example 2.21. Consider its structure map as partial, so that it becomes an $\bar{H}$-algebra. As such, it is a λ-CIA—in fact, the unique solution of an M-equation morphism $e : X \to H_\Sigma X + FY + \{\bot\}$ gives its operational semantics, i. e., each variable in X is mapped to the tree unfolding of its recursive definition if this unfolding is finite, and to $\bot$ otherwise.

2. Analogously, FY becomes a λ-CIA for the distributive law λ of H_Σ over $\mathcal{P}$ of Example 6.14(2). The unique solution of $e : X \to H_\Sigma X + \mathcal{P}FY$ assigns to a variable x the set of all possible tree unfoldings (taking into account that $e(x') \subseteq FY$ for some variables x') of the recursive definition of x if all these unfoldings are finite and $\emptyset$ else. For example, for the signature with one binary operation symbol $*$ the system

$$x = x_1 * x_2 \qquad x' = x' * x_2 \qquad x_1 = \{\ \overset{*}{\underset{y_1}{/}}\ \overset{}{\underset{y_2}{\backslash}}\ , y_3\} \qquad x_2 = \{\ \overset{*}{\underset{y_3}{/}}\ \overset{}{\underset{y_4}{\backslash}}\ \}$$

has the unique solution with $e^\dagger(x)$ given by the set of trees with elements

and with $e^\dagger(x') = \emptyset$.

3. Let H be an analytic functor and let λ be the distributive law over one of the monads $M = \mathrm{Id} + 1$, $\mathcal{P}$ or $\mathcal{D}$ according to Theorem 6.19. Then the initial H-algebra $i : HI \to I$ exists (since H is finitary) and $Ji : HI \to MI$ is a λ-CIA. In fact, we prove in Section 7.2 that I is the initial λ-CIA.

For the following examples of unary λ-CIAs recall Definition 2.31.

Examples 7.11. Here we consider $H = \mathrm{Id}$ and $\lambda = \mathrm{id} : M \to M$. From Remark 7.6 we see that $a : A \to MA$ is a λ-CIA iff $\mu_A \cdot Ma : MA \to MA$ is a CIA for H, i. e., iff $\mu_A \cdot Ma$ has a unique fixed point a_0 and $MA \setminus \{a_0\}$ has a well-founded order for which $\mu_A \cdot Ma$ is strictly increasing (see Example 2.32(4)).

1. $\eta_A : A \to MA$ is a λ-CIA iff MA is a singleton set since $\mu_A \cdot M\eta_A = \mathrm{id}_{MA}$.

2. For the maybe monad $MX = X + \{\bot\}$, the fixed point of MA must be $\bot$. Thus, $a : A \to A + \{\bot\}$ is a λ-CIA iff A has a well-founded order for which $a : A \to A + \{\bot\}$ is strictly increasing, i. e., $a(b) \neq \bot$ implies $b < a(b)$.

3. For $M = \mathcal{P}$, an $\bar{H}$-algebra $a : A \to \mathcal{P}A$ can be considered as a directed graph with node set A (i. e., a binary relation on A) where there is an edge from v to w iff $v \in a(w)$, and vice versa. Then (A, a) is a λ-CIA iff this graph is well-founded, see Corollary 7.51 below. We could add two equivalent formulations; cf. Example 2.32(4).

4. For the environment monad $MX = X^E$, $a : A \to A^E$ is a λ-CIA iff for each $i \in E$ the map $\pi_i \cdot a : A \to A$ is a unary CIA. In fact, this is easy to see by extending (the appropriately simplified version of) diagram (7.2) by each π_i, see also Theorem 7.53.

5. Let $M = F$ be the free monad on H_Σ assigning to a set X the set of all finite Σ-trees on X where Σ contains operations of arity ≥ 1. Then there are no unary λ-CIAs. In order to see this, assume that $a : A \to MA$ is a λ-CIA. Then $\mu_A \cdot Ma$ has a unique fixed point t.

Observe that the action of $\mu_A \cdot Ma$ is that of replacing in all trees of MA each leaf labeled by $b \in A$ with $a(b)$. This implies that for every leaf of the tree t labeled by an element $b \in A$ we have that $a(b)$ is the single-node tree labeled by b. But then every Σ-tree whose leaves have labels that also appear as leaf labels of t is a fixed point of $\mu_A \cdot Ma$. Hence, t does not contain a leaf labeled in A. On the other hand, each tree with no such leaves is a fixed point of $\mu_A \cdot Ma$. Thus there must be a unique constant in Σ and no other operation symbols, and so M is the maybe monad.

Proposition 7.12. *Every Kleisli-CIA is a λ-CIA.*

Proof. Let $a : HA \to MA$ be a Kleisli-CIA and let $e : X \to HX + MA$ be an M-equation morphism. We form a flat equation morphism $\bar{e} : X \nrightarrow \bar{H}X + A$ as follows:

$$\bar{e} = (X \xrightarrow{Je} \bar{H}X + MA \xrightarrow{\mathrm{id}_{\bar{H}X} + \varepsilon_A} \bar{H}X + A)$$

It is clear from the definitions of solutions (see Definitions 7.2/2.30 and 7.5) that a morphism $s : X \nrightarrow A$ is a solution of $\bar{e}$ in the Kleisli-CIA A iff it is a solution of e. Since the former exists uniquely, so does the latter; thus (A, a) is a λ-CIA. □

The following example demonstrates that the converse of Proposition 7.12 does not hold in general.

Example 7.13. The *finite-list monad* $MX = X^*$ has the unit given by singleton lists and the multiplication by flattening a list of lists. Let $H = \mathrm{Id}$, let $MX = X^*$ and let $\lambda = \mathrm{id} : M \to M$. Consider the algebra $A = \{0, 1\}$ with the structure $a : A \to A^*$ given by $a(0) = [1]$ and $a(1) = [1, 1]$, where the square brackets denote lists. Then (A, a) is a (unary) λ-CIA (cf. Example 7.11); in fact, $\mu_A \cdot Ma$ has as its unique fixed point the empty list, every list starting with 0 is mapped to a list starting with 1, and every list starting with 1 is mapped to a longer list. So an appropriate well-founded order on A^* such that $\mu_A \cdot Ma$ is strictly increasing on nonempty lists is given by putting $v < w$ if either v is shorter than w or the lengths agree and v goes before w lexicographically (this is even a well-order on A^*).

To see that (A, a) is not a Kleisli-CIA, consider the equation $x = [x, 1]$ as a flat equation morphism $e : X \to (X + A)^*$. It is not difficult to check that for a solution $e^\dagger(x) = [a_1, \ldots, a_n]$ we have $[a_1, \ldots, a_n] = a(a_1) \cdot \cdots \cdot a(a_n) \cdot [1]$ where $\cdot$ denotes concatenation of lists. Thus $a_n = 1$, and therefore, since $a(a_n) = [1, 1]$ we have $a_{n-1} = 1$ etc., so that we have $a_i = 1$ for all $i = 1, \ldots, n$. But then the two sides of the above equation are lists of different length, a contradiction. So there is no solution of e in A, and A is no Kleisli-CIA.

The λ-CIAs for the endofunctor H together with the usual $\bar{H}$-algebra homomorphisms form a category, which we denote by $H\text{-}\mathsf{CIA}_\lambda$; and Kleisli-CIAs and $\bar{H}$-algebra homomorphisms form the category $\bar{H}\text{-}\mathsf{CIA}$. From Proposition 7.12 we see that $\bar{H}\text{-}\mathsf{CIA}$ is a full subcategory of $H\text{-}\mathsf{CIA}_\lambda$ which in turn is a full subcategory of the category $\bar{H}\text{-}\mathsf{Alg}$ of all algebras for $\bar{H}$:

$$\bar{H}\text{-}\mathsf{CIA} \hookrightarrow H\text{-}\mathsf{CIA}_\lambda \hookrightarrow \bar{H}\text{-}\mathsf{Alg}\,.$$

Our next result is that our choice of morphisms for λ-CIAs is appropriate. A similar result holds for CIAs, see [Mil05], Proposition 2.3.

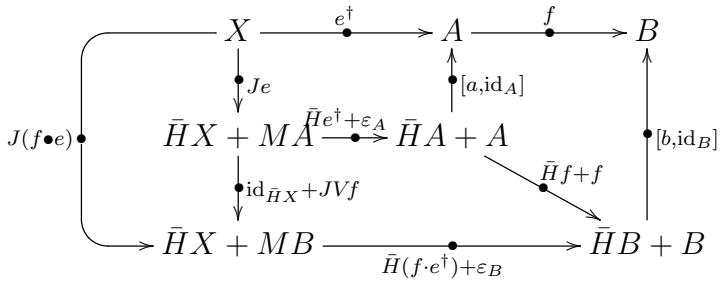
Notation 7.14. For any M-equation morphism $e : X \to HX + MA$ and any morphism $f : A \to MB$ we denote by $f \bullet e$ the M-equation morphism $(\mathrm{id}_{HX} + Vf) \cdot e : X \to HX + MB$ (see Notation 6.4 for V), where the parameters in e are "renamed by f". Here we abuse notation, cf. Notation 2.36.

Proposition 7.15. *Let $f : A \nrightarrow B$ be a morphism, and let $a : \bar{H}A \nrightarrow A$ and $b : \bar{H}B \nrightarrow B$ be λ-CIAs. Then the following statements are equivalent:*

1. *f is an $\bar{H}$-algebra homomorphism from (A, a) to (B, b).*

2. *f preserves solutions, i. e., for all M-equation morphisms $e : X \to HX + MA$ we have $(f \bullet e)^\dagger = f \cdot e^\dagger$ (here $\cdot$ is composition in Set_M).*

Proof. (1) $\Rightarrow$ (2): Consider an $\bar{H}$-algebra homomorphism $f : A \nrightarrow B$ from $a : \bar{H}A \nrightarrow A$ to $b : \bar{H}B \nrightarrow B$. To prove that f preserves solutions we must show that the outside of the diagram

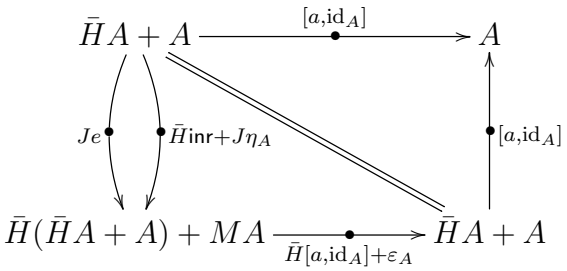


commutes. The right-hand part commutes since f is an $\bar{H}$-algebra homomorphism, and the upper square does since $e^\dagger$ is a solution of the M-equation morphism e. The left-hand part commutes since J preserves composition and coproducts. Finally, consider the lower part componentwise: the left-hand component clearly commutes and the right-hand one does due to the naturality of ε.

(2) $\Rightarrow$ (1): Let f be solution preserving and recall that for the lifting $\bar{H}$ of H we have $\bar{H}J = JH$. Consider the M-equation morphism

$$e = (\, HA + A \xrightarrow{H\mathsf{inr}+\eta_A} H(HA + A) + MA\,).$$

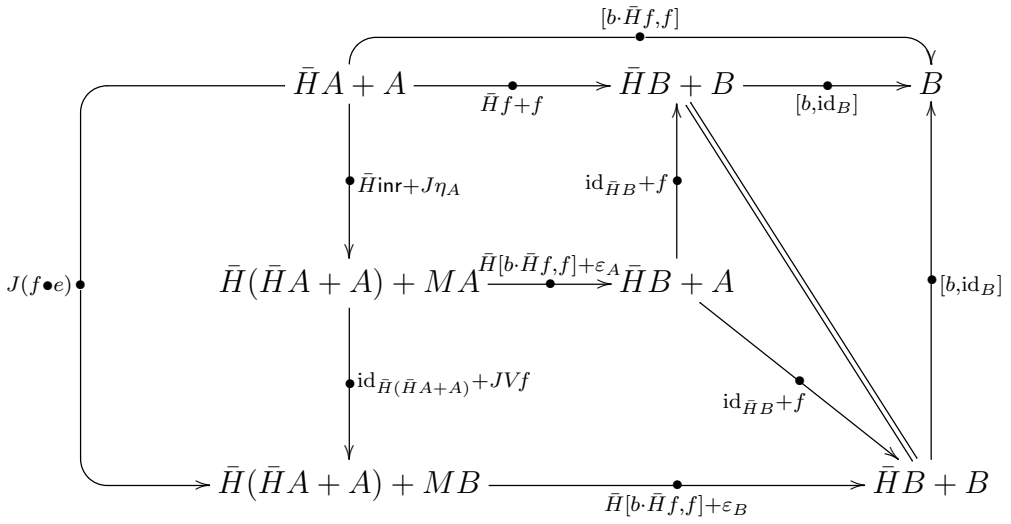
Its unique solution is displayed in the diagram



which commutes: for the equality of the two left-hand arrows use that J preserves coproducts and that $\bar{H}J = JH$, for the nontrivial component of the lower left-hand triangle use that J is identity-on-objects and the adjunction law $\varepsilon_{JA} \cdot J\eta_A = \mathrm{id}_{JA}$, and the remaining triangle is trivial. This proves $e^\dagger = [a, \mathrm{id}_A]$, and since f is solution preserving we get

$$(f \bullet e)^\dagger = f \cdot e^\dagger = [f \cdot a, f]\,. \tag{7.3}$$

Next we show that $[b \cdot \bar{H}f, f] : HA + A \to B$ is a solution of $f \bullet e$. To this end we establish that the diagram



commutes. For the commutativity of the lower part use for the right-hand component the naturality of ε, the left-hand one is trivial. The upper left-hand square commutes by the adjunction law $\varepsilon J \cdot J\eta = \mathrm{id}_J$. The left-hand

part is again due to J preserving coproducts, and the remaining triangles commute obviously. Thus

$$(f \bullet e)^\dagger = [b \cdot \bar{H} f, f] \,. \tag{7.4}$$

We complete the proof by observing that by the uniqueness of solutions of $f \bullet e$, the left-hand components of (7.3) and (7.4) give $f \cdot a = b \cdot \bar{H} f$ as desired. □

7.2 Free Kleisli-CIAs and Free λ-CIAs

The free Kleisli-CIAs and free λ-CIAs for liftings $\bar{H}$ on Set_M differ in their shape, depending on the monad M as well as on the lifting $\bar{H}$. They may even not exist. Thus our approach in this section is as follows: we go through the five monads from Example 6.1 and prove, for suitable classes of liftings and using different techniques, what the free Kleisli-CIAs and free λ-CIAs are. The first three monads $\mathrm{Id} + 1$, $\mathcal{P}$ and $\mathcal{D}$ all have CPO-enriched Kleisli categories which makes it possible to handle them in common. For the monad $\mathcal{P}^+$ we do not succeed in finding free Kleisli-CIAs or free λ-CIAs, but we check possible candidates and prove a result about them we shall need in Chapter 8. We give an overview over the cases investigated in this section:

- monads with CPO-enriched Kleisli categories (including $\mathrm{Id} + 1$, $\mathcal{P}$ and $\mathcal{D}$) and locally monotone liftings,
- $(-)^E$ and canonical liftings, and
- $\mathcal{P}^+$ and canonical liftings.

Monads with CPO-enriched Kleisli Categories and Locally Monotone Liftings

We have already stated in Example 7.10(3) that in a number of concrete cases an initial algebra for the functor H is an initial λ-CIA. In the first part of this section we shall establish that free H-algebras yield free λ-CIAs whenever Set_M is suitably CPO-enriched and the lifting $\bar{H}$ is locally monotone. Our results here are an application of the work of Hasuo, Jacobs and Sokolova [HJS07] and of the work in [Mil05].

Recall that a CPO is a partially ordered set with a least element and with joins of ω-chains. A category $\mathcal{C}$ is *CPO-enriched* if each hom-set carries the structure of a CPO such that composition is continuous (i. e., preserves

joins of ω-chains). Furthermore, recall that an endofunctor H on the CPO-enriched category $\mathcal{C}$ is *locally monotone* (*locally continuous*) if each derived function $\mathcal{C}(X,Y) \to \mathcal{C}(HX,HY)$ is monotone (continuous).

Assumption 7.16. In the first part of this section we assume that

1. H is a finitary endofunctor on Set,
2. λ is a distributive law of H over the monad M (equivalently, $\bar{H}$ is a lifting of H to Set_M),
3. Set_M is CPO-enriched and the composition in Set_M is left-strict, i. e., for each $f : X \nrightarrow Y$ the maps $- \cdot f$ preserve the least element,
4. $\bar{H}$ is locally monotone.

Remark 7.17. 1. Notice that the coproducts of Set_M are CPO-enriched. In fact, it is easy to see that the copairing map

$$[-,-] : \mathsf{Set}_M(X,Z) \times \mathsf{Set}_M(Y,Z) \to \mathsf{Set}_M(X+Y,Z)$$

is continuous (and hence monotone). Clearly, for every object X we have the distributive law $\mathsf{can} \cdot (\lambda + \eta_X) : H_X M \to M H_X$, where $H_X = H(-) + X$. It follows that $\overline{H_X} = \bar{H}_X = \bar{H}(-) + X$ is locally monotone (locally continuous) whenever $\bar{H}$ itself is.

2. Since the functor H is finitary, it has free algebras. We denote by $\phi_X : HFX \to FX$ the structure of a free H-algebra on X and by $\eta_X : X \to FX$ the universal map. Recall from Remark 2.20, items (1) and (2) that a free H-algebra on X is the same as an initial algebra for the functor H_X and that $[\phi_X, \eta_X]$ is an isomorphism. Finally, for the initial H-algebra we use the notation $i : HI \to I$.

Examples 7.18. In this example, we let M be one of the monads $\mathrm{Id} + 1$, $\mathcal{P}$ or $\mathcal{D}$.

1. Assume that P is a finitary polynomial functor; then, as shown in [HJS07], Assumption 7.16 is satisfied. In fact, Set_M is CPO-enriched with a left-strict composition since in all three cases we have a CPO structure $\sqsubseteq$ on each MY: for $\mathrm{Id} + 1$ take the flat CPO structure (i. e., $x \sqsubseteq y$ iff $x = \bot$ or $x = y$), for $\mathcal{P}$ take inclusion, and for $\mathcal{D}$ take the pointwise order ($d \sqsubseteq d'$ iff $d(x) \leq d'(x)$ for all $x \in Y$). This yields a CPO structure $\sqsubseteq$ on $\mathsf{Set}_M(X,Y)$ in a pointwise fashion: $f \sqsubseteq g$ iff for all $x \in X$ we have $f(x) \sqsubseteq g(x)$. It is proved in [HJS07] that the lifting $\bar{P}$ corresponding to the canonical distributive law $\lambda : PM \to MP$ from Lemma 6.12 is locally continuous.

2. The lifting of every analytic functor H corresponding to the canonical distributive law $\lambda : HM \to MH$ from Theorem 6.19 is locally continuous. Indeed, recall that H is the quotient of a polynomial functor P via some $\epsilon : P \to H$ such that equation (6.4) holds. Then it is easy to see that $\bar{H}$ is locally continuous: in fact, due to the naturality of ϵ and equation (6.4) we have for every $f : X \nrightarrow Y$ in Set_M a commutative square (written in Set)

$$\begin{array}{ccccc} & & \overset{\bar{P}f}{\frown} & & \\ PX & \xrightarrow[Pf]{} & PMY & \xrightarrow[\tilde{\lambda}_Y]{} & MPY \\ {\scriptstyle \epsilon_X}\downarrow & & \downarrow{\scriptstyle \epsilon_{MY}} & & \downarrow{\scriptstyle M\epsilon_Y} \\ HX & \xrightarrow{Hf} & HMY & \xrightarrow{\lambda_Y} & MHY \\ & & \underset{\bar{H}f}{\smile} & & \end{array} \tag{7.5}$$

showing that ϵ amounts to a natural transformation $\bar{\epsilon} : \bar{P} \to \bar{H}$ with the components $\bar{\epsilon}_X = J\epsilon_X : \bar{P}X \to \bar{H}X$. Suppose that $(f_n)_{n\in\mathbb{N}}$ is an ω-chain in $\mathsf{Set}_M(X, Y)$. Then we have

$$\begin{aligned} \bar{H}(\bigsqcup f_n) \cdot \bar{\epsilon}_X &= \bar{\epsilon}_Y \cdot \bar{P}(\bigsqcup f_n) && \text{(naturality of } \bar{\epsilon}) \\ &= \bar{\epsilon}_Y \cdot \bigsqcup \bar{P} f_n && \text{(local continuity of } \bar{P}) \\ &= \bigsqcup (\bar{\epsilon}_Y \cdot \bar{P} f_n) && \text{(continuity of } \cdot\,) \\ &= \bigsqcup (\bar{H} f_n \cdot \bar{\epsilon}_X) && \text{(naturality of } \bar{\epsilon}) \\ &= \bigsqcup \bar{H} f_n \cdot \bar{\epsilon}_X && \text{(continuity of } \cdot\,). \end{aligned}$$

Since ϵ_X is a surjective map and the left adjoint J preserves epimorphisms, we can cancel $\bar{\epsilon}_X$ to obtain $\bar{H}(\bigsqcup f_n) = \bigsqcup \bar{H} f_n$, as desired.

Theorem 7.19. *([HJS07], Theorem 3.3) Under Assumption 7.16, the initial H-algebra $i : HI \to I$ yields a final $\bar{H}$-coalgebra $J(i^{-1}) : I \nrightarrow \bar{H}I$ and an initial $\bar{H}$-algebra $Ji : \bar{H}I \nrightarrow I$.*

Theorem 7.20. *Under Assumption 7.16, the free H-algebra $\phi_X : HFX \to FX$ on X with universal map $\eta_X : X \to FX$ yields a free $\bar{H}$-algebra $J\phi_X : \bar{H}FX \nrightarrow FX$ on X with universal map $J\eta_X : X \nrightarrow FX$, and this is a Kleisli-CIA.*

Proof. By Remark 7.17(2), $[\phi_X, \eta_X] : HFX + X \to FX$ is an initial algebra for $H_X = H(-) + X$. Since $\bar{H}_X = \bar{H}(-) + X$ is locally monotone (see Remark 7.17(1)), we can apply Theorem 7.19 to see that $J([\phi_X, \eta_X]^{-1})$ is

a final coalgebra for $\bar{H}_X$ on Set_M. Then from Example 2.32(1) we see that $(FX, J\phi_X)$ is a (free) Kleisli-CIA on X with universal map $J\eta_X$. □

Corollary 7.21. *Under Assumption 7.16, a free H-algebra yields a free Kleisli-CIA and a free λ-CIA.*

Proof. We complete the proof of Theorem 7.20 by observing that every Kleisli-CIA also is a λ-CIA by Proposition 7.12 and the freeness among λ-CIAs follows from freeness among $\bar{H}$-algebras. In fact, Theorem 7.19 also yields that $J[\phi_X, \eta_X] : \bar{H}FX + X \nrightarrow FX$ is an initial algebra, thus a free $\bar{H}$-algebra on X. □

Example 7.22. Let M be one of the monads $\mathrm{Id} + 1$, $\mathcal{P}$ or $\mathcal{D}$. For a finitary polynomial functor H_Σ, the free algebra FX of all finite Σ-trees on X yields a free $\bar{H}_\Sigma$-algebra on X and also a free Kleisli-CIA and a free λ-CIA on X, see Example 2.21.

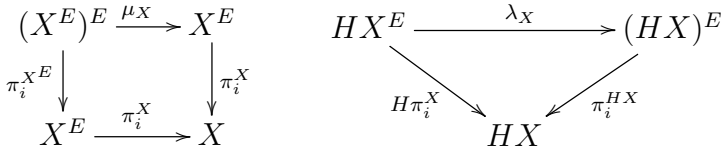
Environment Monad and Canonical Liftings

Next we turn to the environment monad; as we shall see, in this case the free λ-CIAs for canonical liftings of H do not arise from the free H-algebra.

Assumption 7.23. We now assume that

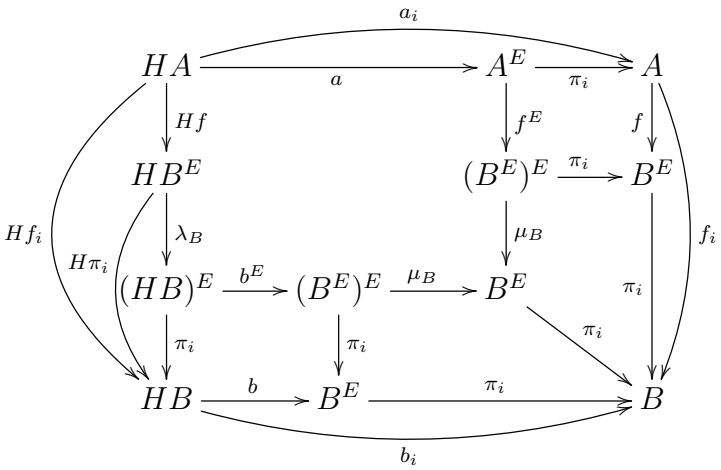
1. H is an iteratable endofunctor on Set (see Definition 2.50),
2. λ is the canonical distributive law of H over the environment monad $M = (-)^E$ from Example 6.23(2) (equivalently, $\bar{H}$ is the canonical lifting of H to Set_M).

We first prove a lemma fundamental for working with the environment monad and canonical liftings. Recall from Examples 6.1(5) and 6.23(2) that the multiplication μ_X of the environment monad and the canonical distributive law λ_X make the following squares commute for every $i \in E$:



Lemma 7.24. *An $\bar{H}$-algebra is, equivalently, a pair $(A, (a_i)_{i\in E})$ where each $a_i : HA \to A$ is an H-algebra. Also, an $\bar{H}$-algebra homomorphism from $(A, (a_i)_{i\in E})$ to $(B, (b_i)_{i\in E})$ is, equivalently, a family $(f_i)_{i\in E}$ of H-algebra homomorphisms from (A, a_i) to (B, b_i), $i \in E$.*

Proof. Given an $\bar{H}$-algebra $a : HA \to A^E$, we put $a_i = \pi_i \cdot a : HA \to A$, and, conversely, every family $(a_i)_{i\in E} : HA \to A$ induces a unique $a : HA \to A^E$. Also, we shall now prove that every $\bar{H}$-algebra homomorphism f from (A, a) to (B, b) induces a family $(f_i)_{i\in E}$ of homomorphisms, and conversely. Indeed, given the homomorphism f, big inner square in the diagram



commutes. Then the outside commutes for every $i \in E$, and so $(\pi_i \cdot f)_{i\in E}$ is the desired family. Conversely, the family $(f_i)_{i\in E}$ such that the outside commutes for each $i \in E$ induces the unique morphism f into the product B^E, and then the inner square commutes. □

Corollary 7.25. *Under Assumption 7.23, the final coalgebra TX for $H(-)+X$ yields a free λ-CIA (and a free Kleisli-CIA) on X. More precisely, the inverse of the coalgebra structure yields an H-algebra structure $\tau_X : HTX \to TX$ and a map $\eta_X : X \to TX$, and $(TX, J\tau_X)$ is a free λ-CIA for H (and a free Kleisli-CIA for $\bar{H}$) with universal arrow $J\eta_X$.*

Proof. Since (TX, τ_X) is a (free) CIA for H by Corollary 2.35, it follows from Theorem 7.53 which we shall prove below that $(TX, J\tau_X)$ is a Kleisli-CIA and a λ-CIA.

The freeness follows from the facts that, firstly, again TX is a free CIA on X for H with universal arrow η_X, and that, secondly, $\bar{H}$-algebras are families $(a_i : HA \to A)_{i\in E}$ of H-algebras with the same carrier A, and similarly, $\bar{H}$-algebra homomorphisms are E-indexed families of H-algebra homomorphisms (see Lemma 7.24). □

Example 7.26. In case of a polynomial functor H_Σ the free λ-CIA and free Kleisli-CIA on X for the canonical lifting $\bar{H}_\Sigma$ is carried by the set TX of all finite and infinite Σ-trees on X, see Example 2.29.

For the special case $E = 1$ the environment monad just is the identity monad $M = \mathrm{Id}$. The canonical distributive laws of every endofunctor H over M are given by $\lambda = \mathrm{id}$, see Example 6.23(2). Here we have $J = \mathrm{Id}$, thus Corollary 7.25 states that the free λ-CIAs and Kleisli-CIAs are given by the free CIAs. This is in accordance with the observation (from Remarks 7.6 and 7.3) that for those distributive laws λ-CIAs, Kleisli-CIAs and ordinary CIAs are all the same, or, more precisely, the categories $H\text{-}\mathsf{CIA}_\lambda$ and $\bar{H}\text{-}\mathsf{CIA}$ coincide with the category $H\text{-}\mathsf{CIA}$.

Nonempty Powerset Monad and Canonical Liftings

For the nonempty powerset monad $\mathcal{P}^+$ neither the free H-algebra FY on Y nor the final $(H(-) + Y)$-coalgebra TY yields a free λ-CIA. We give corresponding counterexamples which show that these are not even λ-CIAs:

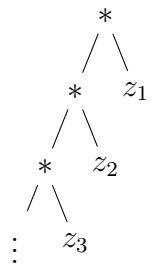
Example 7.27. Consider the identity functor $H = \mathrm{Id}$ and its canonical distributive law $\lambda = \mathrm{id}$ over $\mathcal{P}^+$. Assume that $J\phi_Y = \eta^+_{FY} \cdot \phi_Y : FY \to \mathcal{P}^+FY$ is a λ-CIA. Then, as explained in Example 7.11, $\mu^+_{FY} \cdot \mathcal{P}^+(\eta^+_{FY} \cdot \phi_Y) = \mathcal{P}^+\phi_Y : \mathcal{P}^+\mathrm{Id}F \to \mathcal{P}^+F$ must have a unique fixed point. But since it maps every nonempty set of trees to a different set (the minimum tree depth is increased by 1), there is no such fixed point. Thus $J\phi_Y$ is no λ-CIA.

Example 7.28. Consider the polynomial functor $HX = X \times X$ corresponding to the signature with one binary operation $*$ and the canonical distributive law over $\mathcal{P}^+$ from Example 6.14(4). Assume that $J\tau_Y = \eta^+_{TY} \cdot \tau_Y : HTY \to \mathcal{P}^+TY$ is a λ-CIA. Then, according to Remark 7.6, $\mu^+_{TY} \cdot \mathcal{P}^+(\eta^+_{TY} \cdot \tau_Y) \cdot \lambda_{TY} = \mathcal{P}^+\tau_Y \cdot \lambda_{TY} : H\mathcal{P}^+TY \to \mathcal{P}^+TY$ is an ordinary CIA for H. To reach a contradiction and thus the conclusion that $J\tau_Y$ is no λ-CIA, we show that in general the H-algebras $\mathcal{P}^+\tau_Y \cdot \lambda_{TY}$ are no CIAs. More precisely, we show that solutions in the H-algebras $\mathcal{P}^+\tau_Y \cdot \lambda_{TY}$ need not be unique: consider some nonempty set Y, so we have at least two different trees $t_1 \neq t_2$ in TY. We form the flat equation morphism $e : X \to HX + \mathcal{P}^+TY$ by giving the following system of recursive equations where $X = \{x, y\}$:

$$\begin{aligned} x &= x * y \\ y &= \{t_1, t_2\} \end{aligned}$$

It has several solutions that differ in what is assigned to the variable x (clearly y is always solved to $\{t_1, t_2\}$): according to Proposition 7.33 below, there is

a greatest solution which is given by the set $e^\dagger(x)$ of all trees



(7.6)

with $z_n \in \{t_1, t_2\}$ for all $n \geq 1$. It is easily checked that this is indeed a solution, i. e. that the square (7.8) (in the proof of Proposition 7.33 below) commutes. Moreover, every tree in every solution must be (7.6) with $z_n \in \{t_1, t_2\}$ for all $n \geq 1$ which is easily proved by induction, so this is indeed the greatest solution. Another solution is given by the set $s_1(x)$ of all trees (7.6) where finitely many z_n are taken to be t_2 (and all others to be t_1). Again (7.8) commutes since adding a new root and one child tree t_1 or t_2 gives again all the trees (7.6) where finitely many z_n are taken to be t_2. This clearly is a proper subset of $e^\dagger(x)$ since for example the tree where all z_n are taken to be t_2 is no longer in $s_1(x)$. Let us consider a third solution given by the set $s_2(x)$ of all trees (7.6) where finitely many z_n are taken to be t_1 (and all others to be t_2). The arguments that it is a solution different from $e^\dagger$ are symmetric to the ones for s_1. Comparing s_1 and s_2, we see that these solutions are also different from each other, in fact we have $s_1(x) \cap s_2(x) = \emptyset$. This implies that in general there exists no smallest solution since a solution must assign to every variable a nonempty set.

In fact, we do not know in general what the free λ-CIA is or whether it exists. In Example 7.28 we showed that $J\tau_Y$ is no λ-CIA by proving $\mathcal{P}^+\tau_Y \cdot \lambda_{TY}$ to be no CIA. However, we next prove (in Proposition 7.36) that $\mathcal{P}^+\tau_Y \cdot \lambda_{TY}$ is a *complete Elgot algebra* (or *CEA*, for short, but not the free one in general), since we shall exploit this result in Chapter 8 in order to prove Proposition 8.12.

Assumption 7.29. Here we assume that

1. H is a finitary polynomial endofunctor on Set,

2. λ is the canonical distributive law of H over the nonempty powerset monad $M = \mathcal{P}^+$ from Example 6.14(4) (equivalently, $\bar{H}$ is the canonical lifting of H to Set_M).

In the following definitions, we prepare the proof of Proposition 7.33, where we shall see that taking the solutions of all determinizations of a (nonempty) nondeterministic equation e yields a greatest solution of e. The first definition formalizes the "determinizations" of e; the second one shows how a solution of e is assembled from the solutions of the determinizations.

Definition 7.30. Let $e : X \to HX + \mathcal{P}^+Z$ be a flat equation morphism with parameters in $\mathcal{P}^+Z$ and let $f : \bar{X} \to X$ be a function. A flat equation morphism $\bar{e} : \bar{X} \to H\bar{X} + Z$ with parameters in Z is said to be *over e via f*, if $((Hf + Z) \cdot \bar{e})(\bar{x}) \in (\mathsf{can} \cdot (\eta^+_{HX} + \mathcal{P}^+Z) \cdot e \cdot f)(\bar{x})$ for all $\bar{x} \in \bar{X}$.

Definition 7.31. Let Y be any set. For every flat equation morphism $\bar{e} : \bar{X} \to H\bar{X} + TY$ with parameters in TY, we denote by $\bar{e}^\dagger : \bar{X} \to TY$ its unique solution in the free H-CIA, i. e. $\bar{e}^\dagger$ is the unique map making the diagram

$$\begin{array}{ccc} \bar{X} & \xrightarrow{\quad\bar{e}^\dagger\quad} & TY \\ {\scriptstyle\bar{e}}\downarrow & & \uparrow{\scriptstyle[\tau_Y, TY]} \\ H\bar{X} + TY & \xrightarrow[H\bar{e}^\dagger + TY]{} & HTY + TY \end{array} \tag{7.7}$$

commute. We define the operation $(-)^\dagger$ which maps flat equation morphisms $e : X \to HX + \mathcal{P}^+TY$ to morphisms $e^\dagger : X \to \mathcal{P}^+TY$ by

$$e^\dagger(x) = \{\bar{e}^\dagger(\bar{x}) \mid \bar{e} \text{ is over } e \text{ via } f \text{ and } f(\bar{x}) = x\}\,.$$

Remark 7.32. Notice that $(-)^\dagger$ is well-defined since the sets $e^\dagger(x)$ are nonempty: there always exists a flat equation morphism $\bar{e}$ over e. To see this, choose

$$\bar{X} = X, \quad f = \mathrm{id} \quad \text{and} \quad \bar{e}(x) = \begin{cases} e(x) & e(x) \in HX \\ t \in e(x) & e(x) \in \mathcal{P}^+TY\,. \end{cases}$$

Observe in the definition of $\bar{e}$ that since every set $e(x) \in \mathcal{P}^+TY$ is nonempty, there always exists a tree $t \in e(x)$. Then clearly $\bar{e}$ is over e via f.

In the following Proposition, we need to compare solutions of flat equation morphisms. To be able to do so, we now define a partial order on all hom-sets $\mathsf{Set}(X, \mathcal{P}^+Z)$; moreover, this order nearly is a $\mathrm{CPO}^{\mathrm{op}}$ (having a greatest element) except that in some cases meets of descending ω-chains may not exist.
We start by observing that for all sets $Z \neq \emptyset$, $\mathcal{P}^+Z$ carries the partial order $\subseteq$ with greatest element $\top = Z$. For every descending ω-chain $Z_0 \supseteq Z_1 \supseteq Z_2 \supseteq \cdots$ with $Z_n \in \mathcal{P}^+Z$ for all $n \in \mathbb{N}$, its meet is given by $\bigcap_{n\in\mathbb{N}} Z_n \in \mathcal{P}^+Z$

provided that this intersection is nonempty.
This structure lifts pointwise to the hom-sets $\mathsf{Set}(X, \mathcal{P}^+Z)$ with $Z \neq \emptyset$: we define the partial order $\leq$ by $f \leq g \Leftrightarrow \forall x \in X : f(x) \subseteq g(x)$, and the greatest element $const_\top$ by $const_\top(x) = \top$ for all $x \in X$. For every descending ω-chain $f_0 \geq f_1 \geq f_2 \geq \cdots$ with $f_n \in \mathsf{Set}(X, \mathcal{P}^+Z)$ for all $n \in \mathbb{N}$, its meet $\bigwedge_{n\in\mathbb{N}} f_n$ is given by $(\bigwedge_{n\in\mathbb{N}} f_n)(x) = \bigcap_{n\in\mathbb{N}} f_n(x)$ for all $x \in X$ provided that the intersection $\bigcap_{n\in\mathbb{N}} f_n(x)$ is nonempty for every $x \in X$.
In the sequel, we shall work with $Z = TY$ which is the empty set iff the signature Σ corresponding to the polynomial functor H is empty and $Y = \emptyset$. This is an easy to handle (and uninteresting) special case.

The first step in order to prove $J\tau_Y$ to be a CEA is to exhibit a solution assignment. To show that the operation $(-)^\dagger$ from Definition 7.31 assigns solutions to $\mathcal{P}^+$-equation morphisms (or equivalently, flat equation morphisms with parameters in $\mathcal{P}^+TY$) e, one relies on the fact that $e^\dagger$ is built from the solutions of the determinizations of e. This is the overall idea behind the technical proof of the following

Proposition 7.33. *For every set Y, the operation $(-)^\dagger$ from Definition 7.31 assigns to every flat equation morphism its greatest solution in the algebra $\mathcal{P}^+\tau_Y \cdot \lambda_{TY} : H\mathcal{P}^+TY \to \mathcal{P}^+TY$.*

Proof. Given any set Y and any flat equation morphism $e : X \to HX + \mathcal{P}^+TY$ with parameters in $\mathcal{P}^+TY$, we must prove that $e^\dagger$ as in Definition 7.31 is the greatest solution of e in the H-algebra $\mathcal{P}^+\tau_Y \cdot \lambda_{TY}$.

(1) $e^\dagger$ is a solution, i. e., the diagram

$$\begin{array}{ccc} X & \xrightarrow{\quad e^\dagger \quad} & \mathcal{P}^+TY \\ & & \uparrow{\scriptstyle [\mathcal{P}^+\tau_Y, \mathcal{P}^+TY]} \\ {\scriptstyle e}\downarrow & & \mathcal{P}^+HTY + \mathcal{P}^+TY \\ & & \uparrow{\scriptstyle \lambda_{TY}+\mathcal{P}^+TY} \\ HX + \mathcal{P}^+TY & \xrightarrow[He^\dagger+\mathcal{P}^+TY]{} & H\mathcal{P}^+TY + \mathcal{P}^+TY \end{array} \tag{7.8}$$

commutes. We consider it elementwise, i. e. we prove

$$e^\dagger(x) = ([\mathcal{P}^+\tau_Y, \mathcal{P}^+TY] \cdot (\lambda_{TY} + \mathcal{P}^+TY) \cdot (He^\dagger + \mathcal{P}^+TY) \cdot e)(x)$$

for every $x \in X$. We prove the two subset inclusions:

$\subseteq$ Here we assume $t \in e^\dagger(x)$, i. e. there exist $\bar{e} : \bar{X} \to H\bar{X} + TY$, $f : \bar{X} \to X$ and $\bar{x} \in \bar{X}$ such that $\bar{e}$ is over e via f, $f(\bar{x}) = x$ and $t = \bar{e}^\dagger(\bar{x})$. Using diagram (7.7), the latter equation expands to $t = ([\tau_Y, TY] \cdot (H\bar{e}^\dagger + TY) \cdot \bar{e})(\bar{x})$.

We distinguish two cases:
(a) $\bar{e}(\bar{x}) \in H\bar{X}$. In this case we have $\bar{e}(\bar{x}) = \sigma(\bar{x}_1, \ldots, \bar{x}_n)$ for some operation symbol σ from the signature corresponding to the polynomial functor H and for some $\bar{x}_1, \ldots, \bar{x}_n \in \bar{X}$. Let $x_1 = f(\bar{x}_1), \ldots, x_n = f(\bar{x}_n)$. Since $\bar{e}$ is over e via f, we know

$$\begin{aligned}\sigma(x_1, \ldots, x_n) &= \sigma(f(\bar{x}_1), \ldots, f(\bar{x}_n)) \\ &= ((Hf + TY) \cdot \bar{e})(\bar{x}) \\ &\in (\mathsf{can} \cdot (\eta^+_{HX} + \mathcal{P}^+TY) \cdot e \cdot f)(\bar{x}) \\ &= (\mathsf{can} \cdot (\eta^+_{HX} + \mathcal{P}^+TY) \cdot e)(x)\end{aligned}$$

and since $\sigma(x_1, \ldots, x_n) \in HX$ this implies $\sigma(x_1, \ldots, x_n) = e(x)$. Moreover, from

$$\begin{aligned}t &= ([\tau_Y, TY] \cdot (H\bar{e}^\dagger + TY) \cdot \bar{e})(\bar{x}) \\ &= (\tau_Y \cdot H\bar{e}^\dagger)(\sigma(\bar{x}_1, \ldots, \bar{x}_n)) \\ &= \tau_Y(\sigma(\bar{e}^\dagger(\bar{x}_1), \ldots, \bar{e}^\dagger(\bar{x}_n)))\end{aligned}$$

we see that t has σ as its root label and n child trees $t_1, \ldots, t_n$ given by $\bar{e}^\dagger(\bar{x}_1), \ldots, \bar{e}^\dagger(\bar{x}_n)$. So clearly for every $1 \leq i \leq n$ we have

$$t_i = \bar{e}^\dagger(\bar{x}_i) \in \{\bar{e}^\dagger(\bar{x}_i) \mid \bar{e} \text{ is over } e \text{ via } f \text{ and } f(\bar{x}_i) = x_i\} = e^\dagger(x_i)\,.$$

From this and the action of λ described in Example 6.14(4) we finally derive

$$\begin{aligned}t &= \tau_Y(\sigma(t_1, \ldots, t_n)) \\ &\in (\mathcal{P}^+\tau_Y \cdot \lambda_{TY})(\sigma(e^\dagger(x_1), \ldots, e^\dagger(x_n))) \\ &= (\mathcal{P}^+\tau_Y \cdot \lambda_{TY} \cdot He^\dagger)(\sigma(x_1, \ldots, x_n))\end{aligned}$$

which implies $t \in ([\mathcal{P}^+\tau_Y, \mathcal{P}^+TY] \cdot (\lambda_{TY} + \mathcal{P}^+TY) \cdot (He^\dagger + \mathcal{P}^+TY) \cdot e)(x)$ as desired.
(b) $\bar{e}(\bar{x}) \in TY$. In this case we have $\bar{e}(\bar{x}) = t$. Since $\bar{e}$ is over e via f, see Definition 7.30, we know

$$\begin{aligned}t &= ((Hf + TY) \cdot \bar{e})(\bar{x}) \\ &\in (\mathsf{can} \cdot (\eta^+_{HX} + \mathcal{P}^+TY) \cdot e \cdot f)(\bar{x}) \\ &= (\mathsf{can} \cdot (\eta^+_{HX} + \mathcal{P}^+TY) \cdot e)(x)\end{aligned}$$

and since $t \in TY$ this implies $t \in e(x)$. Moreover, $t \in ([\mathcal{P}^+\tau_Y, \mathcal{P}^+TY] \cdot (\lambda_{TY} + \mathcal{P}^+TY) \cdot (He^\dagger + \mathcal{P}^+TY) \cdot e)(x)$ holds as desired.

$\supseteq$ Here we assume $t \in ([\mathcal{P}^+\tau_Y, \mathcal{P}^+TY] \cdot (\lambda_{TY} + \mathcal{P}^+TY) \cdot (He^\dagger + \mathcal{P}^+TY) \cdot e)(x)$. We distinguish two cases:
(a) $e(x) \in HX$. In this case we have $e(x) = \sigma(x_1, \ldots, x_n)$ and thus $t \in (\mathcal{P}^+\tau_Y \cdot \lambda_{TY} \cdot He^\dagger)(\sigma(x_1, \ldots, x_n)) = (\mathcal{P}^+\tau_Y \cdot \lambda_{TY})(\sigma(e^\dagger(x_1), \ldots, e^\dagger(x_n)))$, i. e., taking the action of λ described in Example 6.14(4) into account, t has root label σ and child trees $t_1 \in e^\dagger(x_1), \ldots, t_n \in e^\dagger(x_n)$. This means that for every $1 \leq i \leq n$ we have $t_i = \bar{e}_i^\dagger(\bar{x}_i)$ for some $\bar{e}_i : \bar{X}_i \to H\bar{X}_i + TY$ over $e : X \to HX + \mathcal{P}^+TY$ via $f_i : \bar{X}_i \to X$ with $f_i(\bar{x}_i) = x_i$. We form $\bar{X} = \bar{X}_1 + \cdots + \bar{X}_n + \{\bar{x}\}$ and define $f : \bar{X} \to X$ by

$$f(y) = \begin{cases} x & \text{if } y = \bar{x} \\ f_i(y) & \text{if } y \in \bar{X}_i \end{cases}$$

and $\bar{e} : \bar{X} \to H\bar{X} + TY$ by

$$\bar{e}(y) = \begin{cases} \sigma(\bar{x}_1, \ldots, \bar{x}_n) & \text{if } y = \bar{x} \\ \bar{e}_i(y) & \text{if } y \in \bar{X}_i \,. \end{cases}$$

To verify that $\bar{e}$ is over e via f we only need to check

$$\begin{aligned} ((Hf + TY) \cdot \bar{e})(\bar{x}) &= Hf(\sigma(\bar{x}_1, \ldots, \bar{x}_n)) \\ &= \sigma(f(\bar{x}_1), \ldots, f(\bar{x}_n)) \\ &= \sigma(f_1(\bar{x}_1), \ldots, f_n(\bar{x}_n)) \\ &= \sigma(x_1, \ldots, x_n) \\ &= e(x) \\ &= (e \cdot f)(\bar{x}) \\ &\in (\mathsf{can} \cdot (\eta^+_{HX} + \mathcal{P}^+TY) \cdot e \cdot f)(\bar{x}) \end{aligned}$$

since for all other $y \in \bar{X}$ this follows from the $\bar{e}_i$ being over e via f_i. Since the injections $\in_i$: $\bar{X}_i \to \bar{X}$ are H-coalgebra homomorphisms between $\bar{e}_i$ and $\bar{e}$ by the definition of $\bar{e}$, the left-hand parts of the diagrams

$$\begin{array}{ccccc} \bar{X}_i & \xrightarrow{\in_i} & \bar{X} & \xrightarrow{\bar{e}^\dagger} & TY \\ {\scriptstyle \bar{e}_i}\downarrow & & {\scriptstyle \bar{e}}\downarrow & & \uparrow{\scriptstyle [\tau_Y, TY]} \\ H\bar{X}_i + TY & \xrightarrow[H\in_i + TY]{} & H\bar{X} + TY & \xrightarrow[H\bar{e}^\dagger + TY]{} & HTY + TY \end{array}$$

commute, and the right-hand part commutes since $\bar{e}^\dagger$ is the unique solution of $\bar{e}$. So the outside commutes, and since τ_Y is a CIA we conclude $\bar{e}_i^\dagger = \bar{e}^\dagger \cdot \in_i$

for the unique solution of $\bar{e}_i$. In particular this means $\bar{e}_i^\dagger(\bar{x}_i) = \bar{e}^\dagger(\bar{x}_i)$ which we use to verify that

$$\begin{aligned} t &= \tau_Y(\sigma(t_1, \ldots, t_n)) \\ &= \tau_Y(\sigma(\bar{e}_1^\dagger(\bar{x}_1), \ldots, \bar{e}_n^\dagger(\bar{x}_n))) \\ &= \tau_Y(\sigma(\bar{e}^\dagger(\bar{x}_1), \ldots, \bar{e}^\dagger(\bar{x}_n))) \\ &= (\tau_Y \cdot H\bar{e}^\dagger)(\sigma(\bar{x}_1, \ldots, \bar{x}_n)) \\ &= ([\tau_Y, TY] \cdot (H\bar{e}^\dagger + TY) \cdot \bar{e})(\bar{x}) \\ &= \bar{e}^\dagger(\bar{x})\,. \end{aligned}$$

Finally, since $\bar{e}$ is over e via f and $f(\bar{x}) = x$, we know that $t \in e^\dagger(x)$ as desired.
(b) $e(x) \in \mathcal{P}^+TY$. In this case we have $t \in e(x)$. We form $\bar{X} = \{\bar{x}\}$ and define $f(\bar{x}) = x$ and $\bar{e}(\bar{x}) = t$. To verify that $\bar{e}$ is over e via f we only need to check

$$((Hf + TY) \cdot \bar{e})(\bar{x}) = t \in e(x) = (e \cdot f)(\bar{x})$$

from which it follows that $((Hf+TY)\cdot\bar{e})(\bar{x}) \in (\mathsf{can}\cdot(\eta^+_{HX}+\mathcal{P}^+TY)\cdot e\cdot f)(\bar{x})$. Since $\bar{e}(\bar{x}) \in TY$ we conclude $\bar{e}^\dagger(\bar{x}) = \bar{e}(\bar{x}) = t$ from diagram (7.7). Finally, since $\bar{e}$ is over e via f and $f(\bar{x}) = x$, we know that $t \in e^\dagger(x)$ as desired.

(2) $e^\dagger$ is the greatest solution. To show this, suppose that $s : X \to \mathcal{P}^+TY$ is any solution of e; we prove $s \leq e^\dagger$ elementwise, i. e. for every $x \in X$ we show $s(x) \subseteq e^\dagger(x)$.

Assume $t \in s(x)$. To show that $t \in e^\dagger(x)$, we have to define some $\bar{e} : \bar{X} \to H\bar{X} + TY$ over e via $f : \bar{X} \to X$ with $f(\bar{x}) = x$ and $t = \bar{e}^\dagger(\bar{x})$. To this end, we use induction over the natural numbers; we start with an empty set of variables and add new variables in each induction step for which we define f and a map $\bar{s} : \bar{X} \to TY$ immediately and $\bar{e}$ one step later.
Base case ($m = 0$): we add one variable $\bar{x}$ to $\bar{X}$ and define $f(\bar{x}) = x$ and $\bar{s}(\bar{x}) = t$.
Induction precondition: for all $\bar{y} \in \bar{X}$ defined in step m, $u = \bar{s}(\bar{y})$ is a subtree of t rooted at level m such that $u \in s(y)$ where $y = f(\bar{y})$.
Induction step ($m \to m + 1$): For every $\bar{y} \in \bar{X}$ defined in step m we have some $y = f(\bar{y})$ and distinguish two cases:
(a) $e(y) \in HX$. Then we have $e(y) = \sigma(y_1, \ldots, y_n)$ and add fresh variables $\bar{y}_1, \ldots, \bar{y}_n$ to $\bar{X}$; we define $\bar{e}(\bar{y}) = \sigma(\bar{y}_1, \ldots, \bar{y}_n)$. Furthermore we define $f(\bar{y}_i) = y_i$ for all $1 \leq i \leq n$. Let $u = \bar{s}(\bar{y})$ and observe that from $u \in s(y)$ it

follows, since s is a solution,

$$\begin{aligned} u &\in ((\mathcal{P}^+\tau_Y + \mathcal{P}^+TY) \cdot (\lambda_{TY} + \mathcal{P}^+TY) \cdot (Hs + \mathcal{P}^+TY) \cdot e)(y) \\ &= (\mathcal{P}^+\tau_Y \cdot \lambda_{TY} \cdot Hs)(\sigma(y_1, \ldots, y_n)) \\ &= (\mathcal{P}^+\tau_Y \cdot \lambda_{TY})(\sigma(s(y_1), \ldots, s(y_n))) \end{aligned}$$

which means that u has root operation symbol σ and child trees $u_1 \in s(y_1), \ldots, u_n \in s(y_n)$. From u being a subtree of t rooted at level m we conclude that the $u_1, \ldots, u_n$ are subtrees of t rooted at level $m+1$, and we define $\bar{s}(\bar{y}_i) = u_i$ for all $1 \leq i \leq n$.
(b) $e(y) \in \mathcal{P}^+TY$. Then we add no variables to $\bar{X}$; only for $u = \bar{s}(\bar{y})$ we define $\bar{e}(\bar{y}) = u$.
This induction completely defines $\bar{X}$, $\bar{e}$, f and $\bar{s}$. We first show that $\bar{s} = \bar{e}^\dagger$, i. e. that $\bar{s}$ is the unique solution of $\bar{e}$ in the CIA τ_Y. We have to show $\bar{s}(\bar{y}) = ([\tau_Y, TY] \cdot (H\bar{s} + TY) \cdot \bar{e})(\bar{y})$ for every $\bar{y} \in \bar{X}$; let $y = f(\bar{y})$, then we distinguish two cases: if $e(y) \in HX$ we get

$$\begin{aligned} ([\tau_Y, TY] \cdot (H\bar{s} + TY) \cdot \bar{e})(\bar{y}) &= ([\tau_Y, TY] \cdot (H\bar{s} + TY))(\sigma(\bar{y}_1, \ldots, \bar{y}_n)) \\ &= \tau_Y(\sigma(\bar{s}(\bar{y}_1), \ldots, \bar{s}(\bar{y}_n))) \\ &= \tau_Y(\sigma(u_1, \ldots, u_n)) \\ &= u \\ &= \bar{s}(\bar{y})\,, \end{aligned}$$

and if $e(y) \in \mathcal{P}^+TY$ we directly get $([\tau_Y, TY] \cdot (H\bar{s} + TY) \cdot \bar{e})(\bar{y}) = u = \bar{s}(\bar{y})$.
Next we prove that $\bar{e}$ is over e via f; to this end let $y = f(\bar{y})$ and distinguish two cases: if $e(y) \in HX$ we get

$$\begin{aligned} ((Hf + TY) \cdot \bar{e})(\bar{y}) &= Hf(\sigma(\bar{y}_1, \ldots, \bar{y}_n)) \\ &= \sigma(f(\bar{y}_1), \ldots, f(\bar{y}_n)) \\ &= \sigma(y_1, \ldots, y_n) \\ &\in (\mathsf{can} \cdot (\eta^+_{HX} + \mathcal{P}^+TY))(\sigma(y_1, \ldots, y_n)) \\ &= (\mathsf{can} \cdot (\eta^+_{HX} + \mathcal{P}^+TY) \cdot e)(y) \\ &= (\mathsf{can} \cdot (\eta^+_{HX} + \mathcal{P}^+TY) \cdot e \cdot f)(\bar{y})\,, \end{aligned}$$

and if $e(y) \in \mathcal{P}^+TY$ we get

$$\begin{aligned}((Hf + TY) \cdot \bar{e})(\bar{y}) &= u \\ &\in (\mathcal{P}^+\mathsf{inr} \cdot s)(y) \\ &= (\mathcal{P}^+\mathsf{inr} \cdot [\mathcal{P}^+\tau_Y, \mathcal{P}^+TY] \\ &\quad \cdot (\lambda_{TY} + \mathcal{P}^+TY) \cdot (Hs + \mathcal{P}^+TY) \cdot e)(y) \\ &= (\mathsf{can} \cdot (\eta^+_{HX} + \mathcal{P}^+TY) \cdot e)(y) \\ &= (\mathsf{can} \cdot (\eta^+_{HX} + \mathcal{P}^+TY) \cdot e \cdot f)(\bar{y})\,.\end{aligned}$$

Since by definition we have $f(\bar{x}) = x$ and $\bar{e}^\dagger(\bar{x}) = \bar{s}(\bar{x}) = t$, we have proved $t \in e^\dagger(x)$ as desired. □

Next we give a different characterization of the operation $(-)^\dagger$ from Definition 7.31 as a greatest fixed point. It is based on a variant of the dual of Kleene's fixed-point theorem: instead of arbitrary ascending ω-chains in a CPO (with least element), we consider a special class of descending ω-chains whose meets exist in the nearly CPO$^{\mathrm{op}}$'s (with greatest element) on the hom-sets $\mathsf{Set}(X, \mathcal{P}^+Z)$ defined above right before Proposition 7.33.

For every equation morphism $e : X \to HX + \mathcal{P}^+Z$ and H-algebra $a : HZ \to Z$ we define the operator $\Phi_{X,Z}$ on $\mathsf{Set}_{\mathcal{P}^+}(X, Z)$ by the diagram for solutions of e in $\mathcal{P}^+a \cdot \lambda_Z$: given a function $f : X \to \mathcal{P}^+Z$, we set $\Phi(f) = [\mathcal{P}^+a, \mathrm{id}] \cdot (\lambda_Z + \mathrm{id}) \cdot (Hf + \mathrm{id}) \cdot e$. We can rewrite the right-hand side to obtain

$$\Phi(f) = [\mathcal{P}^+a \cdot \bar{H}f, \mathrm{id}] \cdot e \tag{7.9}$$

where $\bar{H}$ is the lifting of H to $\mathsf{Set}_{\mathcal{P}^+}$ via the canonical distributive law λ. To prove that Φ is continuous, we need the following

Lemma 7.34. *Liftings $\bar{H}$ of finitary polynomial endofunctors H on Set to $\mathsf{Set}_{\mathcal{P}^+}$ via the canonical distributive law are locally continuous. And postcomposition with maps $\mathcal{P}^+g$ where g is monomorphic, copairing and precomposition in Set are continuous.*

Proof. Throughout the whole proof, let $f_n : X \to \mathcal{P}^+Y$, $n \in \mathbb{N}$, denote a descending ω-chain $f_0 \geq f_1 \geq f_2 \geq \cdots$.

(1) $\bar{H}$ is locally continuous. Recall that $\bar{H}f = \lambda_Y \cdot Hf$ for morphisms

$f : X \to \mathcal{P}^+Y$. Then we have

$$\begin{aligned}
(\bar{H}(\bigwedge_{n\in\mathbb{N}} f_n))(\sigma(x_1,\ldots,x_m)) &= (\lambda_Y \cdot H(\bigwedge_{n\in\mathbb{N}} f_n))(\sigma(x_1,\ldots,x_m)) \\
&= \lambda_Y(\sigma((\bigwedge_{n\in\mathbb{N}} f_n)(x_1),\ldots,(\bigwedge_{n\in\mathbb{N}} f_n)(x_m))) \\
&= \lambda_Y(\sigma(\bigcap_{n\in\mathbb{N}} f_n(x_1),\ldots,\bigcap_{n\in\mathbb{N}} f_n(x_m))) \\
&\overset{(*)}{=} \bigcap_{n\in\mathbb{N}} \lambda_Y(\sigma(f_n(x_1),\ldots,f_n(x_m))) \\
&= \bigcap_{n\in\mathbb{N}} (\lambda_Y \cdot Hf_n)(\sigma(x_1,\ldots,x_m)) \\
&= \bigcap_{n\in\mathbb{N}} (\bar{H}f_n)(\sigma(x_1,\ldots,x_m)) \\
&= (\bigwedge_{n\in\mathbb{N}} (\bar{H}f_n))(\sigma(x_1,\ldots,x_m))
\end{aligned}$$

for all $\sigma(x_1,\ldots,x_m) \in HX$ where the equality marked by $(*)$ holds indeed: we have

$$\begin{aligned}
& \sigma(y_1,\ldots,y_m) \in \lambda_Y(\sigma(\bigcap_{n\in\mathbb{N}} f_n(x_1),\ldots,\bigcap_{n\in\mathbb{N}} f_n(x_m))) \\
\iff\quad & y_i \in \bigcap_{n\in\mathbb{N}} f_n(x_i) \quad \text{for all } 1 \le i \le m \\
\iff\quad & y_i \in f_n(x_i) \quad \text{for all } 1 \le i \le m \text{ and all } n \in \mathbb{N} \\
\iff\quad & \sigma(y_1,\ldots,y_m) \in \lambda_Y(\sigma(f_n(x_1),\ldots,f_n(x_m))) \quad \text{for all } n \in \mathbb{N} \\
\iff\quad & \sigma(y_1,\ldots,y_m) \in \bigcap_{n\in\mathbb{N}} \lambda_Y(\sigma(f_n(x_1),\ldots,f_n(x_m)))
\end{aligned}$$

where we use the concrete description of λ from Example 6.14(4).

(2) Postcomposition with $\mathcal{P}^+g$ where g is monomorphic is continuous.

Let $g : Y \to Z$ be a monomorphism. Then we have

$$\begin{aligned}(\mathcal{P}^+g \cdot (\bigwedge_{n\in\mathbb{N}} f_n))(x) &= \mathcal{P}^+g(\bigcap_{n\in\mathbb{N}} f_n(x)) \\ &= \{g(y) \mid y \in \bigcap_{n\in\mathbb{N}} f_n(x)\} \\ &= \{g(y) \mid y \in f_n(x) \text{ for all } n \in \mathbb{N}\} \\ &\overset{(*)}{=} \{z \mid z \in (\mathcal{P}^+g \cdot f_n)(x) \text{ for all } n \in \mathbb{N}\} \\ &= \bigcap_{n\in\mathbb{N}} (\mathcal{P}^+g \cdot f_n)(x) \\ &= (\bigwedge_{n\in\mathbb{N}} (\mathcal{P}^+g \cdot f_n))(x)\end{aligned}$$

for all $x \in X$ where the equation marked by $(*)$ holds indeed: the subset inclusion $\subseteq$ is easy to see, and for the other one $\supseteq$ use that g is a monomorphism.

(3) Copairing is continuous. Let $h : Z \to \mathcal{P}^+Y$ be a map. Then we have

$$[\bigwedge_{n\in\mathbb{N}} f_n, h](x) = (\bigwedge_{n\in\mathbb{N}} f_n)(x) = \bigcap_{n\in\mathbb{N}} f_n(x) = \bigcap_{n\in\mathbb{N}} [f_n, h](x) = (\bigwedge_{n\in\mathbb{N}} [f_n, h])(x)$$

for all $x \in X$ and similarly

$$[\bigwedge_{n\in\mathbb{N}} f_n, h](z) = h(z) = \bigcap_{n\in\mathbb{N}} h(z) = \bigcap_{n\in\mathbb{N}} [f_n, h](z) = (\bigwedge_{n\in\mathbb{N}} [f_n, h])(z)$$

for all $z \in Z$.

(4) Precomposition is continuous. Let $d : W \to X$ be a map. Then we have

$$((\bigwedge_{n\in\mathbb{N}} f_n) \cdot d)(w) = (\bigwedge_{n\in\mathbb{N}} f_n)(d(w)) = \bigcap_{n\in\mathbb{N}} (f_n \cdot d)(w) = (\bigwedge_{n\in\mathbb{N}} (f_n \cdot d))(w)$$

for all $w \in W$. □

Corollary 7.35. *For every set Y, the operation $(-)^\dagger$ from Definition 7.31 is equivalently given by*

$$e^\dagger = \bigwedge_{n\in\mathbb{N}} e^\dagger_n$$

where the $e^\dagger_n$ are defined by induction on n as in $e^\dagger_0 = const_\top$ and $e^\dagger_{n+1} = \Phi(e^\dagger_n)$.

Proof. Let $e : X \to HX + \mathcal{P}^+TY$ be a flat equation morphism. In Proposition 7.33 we have already shown that a greatest solution (which is a greatest fixed point of Φ by the definition of Φ) exists. From equation (7.9) for Φ with $a = \tau_Y$ and Lemma 7.34 we conclude, since τ_Y is monomorphic, that the function Φ on $\mathsf{Set}(X, \mathcal{P}^+TY)$ is continuous (and thus monotone). From the dual of the Kleene fixed-point theorem the desired result follows immediately: although we have no $\mathrm{CPO}^{\mathrm{op}}$, the Kleene Theorem can be applied since we know the existence of the greatest fixed point already from Proposition 7.33 (together with the well-definedness shown in Remark 7.32). □

We are now ready to prove

Proposition 7.36. *For the canonical distributive law λ of a finitary polynomial endofunctor H on Set over the monad $\mathcal{P}^+$, the algebra $\mathcal{P}^+\tau_Y \cdot \lambda_{TY} : H\mathcal{P}^+TY \to \mathcal{P}^+TY$ is a complete Elgot algebra for every set Y.*

Proof. We already know from Proposition 7.33 that the operation $(-)^\dagger$ from Definition 7.31 assigns to every equation morphism e a (greatest) solution $e^\dagger$ in $\mathcal{P}^+\tau_Y \cdot \lambda_{TY}$; moreover, from Corollary 7.35 we have $e^\dagger = \bigwedge_{n\in\mathbb{N}} e_n^\dagger$. We use this latter characterization of the operation $(-)^\dagger$ to prove functoriality and compositionality (see Definition 2.37), adapting the proof of Proposition 3.5 from [AMV06a].

(1) The operation $(-)^\dagger$ is functorial. Consider a homomorphism $h : X \to Z$ between the coalgebras $e : X \to HX + \mathcal{P}^+TY$ and $g : Z \to HZ + \mathcal{P}^+TY$, i. e. the left-hand square of the diagram

$$\begin{array}{ccccc} X & \xrightarrow{\quad h \quad} & Z & \xrightarrow{\quad g^\dagger_{n+1} \quad} & \mathcal{P}^+TY \\ {\scriptstyle e}\downarrow & & {\scriptstyle g}\downarrow & & \uparrow{\scriptstyle [\mathcal{P}^+\tau_Y\cdot\lambda_{TY},\mathrm{id}]} \\ HX + \mathcal{P}^+TY & \xrightarrow[Hh+\mathrm{id}]{} & HZ + \mathcal{P}^+TY & \xrightarrow[Hg^\dagger_n+\mathrm{id}]{} & H\mathcal{P}^+TY + \mathcal{P}^+TY \end{array}$$

commutes. We also know that the right-hand square commutes for every $n \in \mathbb{N}$. We clearly have $e_0^\dagger = const_\top = const_\top \cdot h = g_0^\dagger \cdot h$ since composition is left-strict for $const_\top$. And whenever $e_n^\dagger = g_n^\dagger \cdot h$, from the commutative diagram above it follows $e_{n+1}^\dagger = g_{n+1}^\dagger \cdot h$. Thus by induction we have $e_n^\dagger = g_n^\dagger \cdot h$ for all $n \in \mathbb{N}$, and since precomposition is continuous by Lemma 7.34, we get

$$e^\dagger = \bigwedge_{n\in\mathbb{N}} e_n^\dagger = \bigwedge_{n\in\mathbb{N}} (g_n^\dagger \cdot h) = (\bigwedge_{n\in\mathbb{N}} g_n^\dagger) \cdot h = g^\dagger \cdot h\,.$$

(2) The operation $(-)^\dagger$ is compositional. Consider equation morphisms $e : X \to HX + Z$ and $g : Z \to HZ + \mathcal{P}^+TY$ and recall Notation 2.36.

We show $(g \blacksquare e)^\dagger \cdot \mathsf{inl} = (g^\dagger \bullet e)^\dagger$ by proving $(g^\dagger \bullet e)^\dagger \leq (g \blacksquare e)^\dagger_n \cdot \mathsf{inl}$ and $(g \blacksquare e)^\dagger \cdot \mathsf{inl} \leq (g^\dagger \bullet e)^\dagger_n$ for all $n \in \mathbb{N}$. This implies

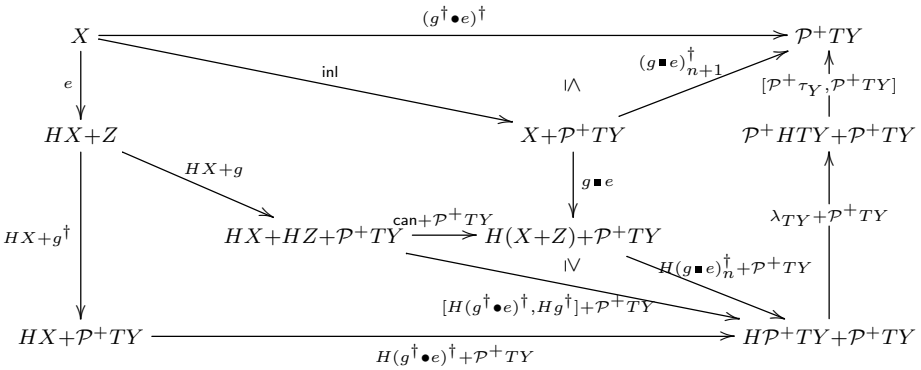
$$(g^\dagger \bullet e)^\dagger \leq \bigwedge_{n \in \mathbb{N}} ((g \blacksquare e)^\dagger_n \cdot \mathsf{inl}) = \bigwedge_{n \in \mathbb{N}} ((g \blacksquare e)^\dagger_n) \cdot \mathsf{inl} = (g \blacksquare e)^\dagger \cdot \mathsf{inl}$$

using continuity of precomposition and also

$$(g \blacksquare e)^\dagger \cdot \mathsf{inl} \leq \bigwedge_{n \in \mathbb{N}} (g^\dagger \bullet e)^\dagger_n = (g^\dagger \bullet e)^\dagger ,$$

thus the desired equation holds. Before we prove both inequalities by induction on n, observe that by the definition of $g \blacksquare e$, inr is a coalgebra homomorphism from g to $g \blacksquare e$, thus functoriality yields $g^\dagger = (g \blacksquare e)^\dagger \cdot \mathsf{inr}$.

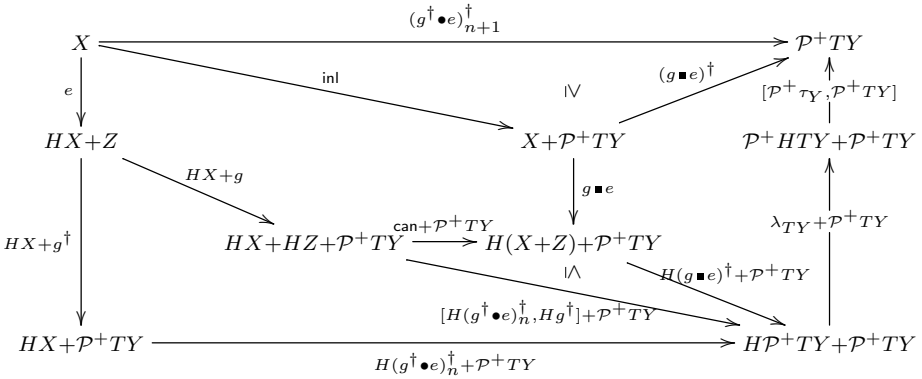
For the first inequality, the base case $(g^\dagger \bullet e)^\dagger \leq const_\top = const_\top \cdot \mathsf{inl} = (g \blacksquare e)^\dagger_0 \cdot \mathsf{inl}$ follows from left-strictness of composition. For the induction step, consider the following diagram:



The outer square commutes since $(g^\dagger \bullet e)^\dagger$ is a solution of $g^\dagger \bullet e$. We want to show the inequality indicated in the upper triangle, so we check that all other inner parts commute: for the parts below the upper triangle, use the definition of $g \blacksquare e$ and the definition of $(g \blacksquare e)^\dagger_{n+1}$. For the little triangle composed with $[\mathcal{P}^+\tau_Y, \mathcal{P}^+TY] \cdot (\lambda_{TY} + \mathcal{P}^+TY)$ we use the induction hypothesis $(g^\dagger \bullet e)^\dagger \leq (g \blacksquare e)^\dagger_n \cdot \mathsf{inl}$ and $g^\dagger = (g \blacksquare e)^\dagger \cdot \mathsf{inr} \leq (g \blacksquare e)^\dagger_n \cdot \mathsf{inr}$ as well as local monotonicity H and monotonicity of coproducts in $\mathsf{Set}_{\mathcal{P}^+}$ which follow from Lemma 7.34. For the lower part the left-hand coproduct component is trivial, and the right-hand component commutes when composed with $[\mathcal{P}^+\tau_Y, \mathcal{P}^+TY] \cdot (\lambda_{TY} + \mathcal{P}^+TY)$ since $g^\dagger$ is a solution of g.

For the second inequality, the base case $(g \blacksquare e)^\dagger \cdot \mathsf{inl} \leq const_\top = (g^\dagger \bullet e)^\dagger_0$ is immediate. For the induction step, consider a similar diagram as for the

first inequality:



The outer square commutes by the definition of $(g^\dagger \bullet e)^\dagger_{n+1}$ and the right-hand square since $(g \blacksquare e)^\dagger$ is a solution of $g \blacksquare e$. The other parts are similar as above, so the desired inequality as indicated in the upper triangle holds. □

7.3 Characterization of Kleisli-CIAs and λ-CIAs

In this section we consider three of our five concrete examples of monads: the maybe monad $M = \mathrm{Id}+1$, the powerset monad $M = \mathcal{P}$ and the environment monad $MX = X^E$. We show that for these monads the Kleisli-CIAs and the λ-CIAs for a finitary polynomial functor H_Σ coincide, where λ is the canonical distributive law of Example 6.14. In fact, in the case of the environment monad we allow arbitrary endofunctors H on Set, where λ is the canonical distributive law of Example 6.23(2). In each case we give a characterization of the Kleisli-CIAs and λ-CIAs.

We first state some results in a more general setting which is common for the maybe and powerset monads. Using these results, we then prove a characterization of λ-CIAs where H_Σ is a finitary polynomial functor and $M = \mathrm{Id}+1$ is the maybe monad and a characterization of λ-CIAs where H_Σ is a finitary polynomial functor and $M = \mathcal{P}$ is the powerset monad separately.

Common Results for Partial and Nondeterministic CIAs

Notation 7.37. For any set A we denote by $!_A : A \to 1$ the unique map from A to the singleton set and by $¡_A : \emptyset \to A$ the unique map from the empty set to A.

Assumption 7.38. Throughout the following part of this chapter let

1. H_Σ be a finitary polynomial functor on Set,

2. (M, η, μ) be a nontrivial monad (i. e., not the monad $(C_1, !, \mathrm{id}_1)$),

3. MA carry a CPO-structure $(\sqsubseteq_A, \bot_A)$, for each set A, such that for the least elements we have $\bot_A = M¡_A(\bot_\emptyset)$,

4. $(\sqsubseteq_{X,A}, \bot_{X,A})$ be the CPO-structure on $\mathsf{Set}(X, MA)$ that arises from the CPO-structure $(\sqsubseteq_A, \bot_A)$ on MA by pointwise definition, for each pair of sets X, A,

5. λ be a distributive law of H_Σ over the monad M which is strict w. r. t. the CPOs $(\sqsubseteq_A, \bot_A)$, i. e., for all A and terms $\sigma(\dots, \bot_A, \dots) \in H_\Sigma MA$ we have
$$\lambda_A(\sigma(\dots, \bot_A, \dots)) = \bot_{H_\Sigma A}\,, \tag{7.10}$$

6. the map (cf. diagram (7.2))
$$F : s \mapsto [a, \mathrm{id}_A] \cdot (\bar{H}_\Sigma s + \varepsilon_A) \cdot Je \tag{7.11}$$
on $\mathsf{Set}(X, MA)$ be continuous w. r. t. the CPO $(\sqsubseteq_{X,A}, \bot_{X,A})$, for each $\bar{H}_\Sigma$-algebra $a : \bar{H}_\Sigma A \nrightarrow A$ and each M-equation morphisms $e : X \to H_\Sigma X + MA$.

Remark 7.39. Assumption 7.38 holds for the maybe and powerset monads $M = \mathrm{Id} + 1$ and $M = \mathcal{P}$: both are nontrivial, and the sets $A + 1$ and $\mathcal{P}A$ carry the flat and subset CPO-structures $\sqsubseteq_A$, respectively. The canonical distributive laws of finitary polynomial functors over both of those monads from Examples 6.14(1) and (2) are easily seen to be strict. And for both monads the map F of (7.11) is continuous since $\bar{H}_\Sigma$ is locally continuous (cf. Assumption 7.16 and Example 7.18(1)).

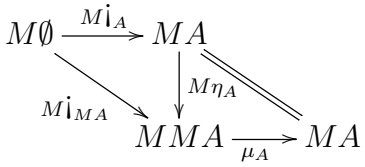
Notation 7.40. From now on we shall drop the subscripts of orders and least elements and simply write $\sqsubseteq$ and $\bot$.

Lemma 7.41. *For any map $f : A \to B$ the map Mf preserves the least element. Furthermore, for all sets A the map μ_A preserves the least element and $\bot \notin \eta_A[A]$.*

Proof. For every $f : A \to B$ initiality of $\emptyset$ makes the left-hand one of the two triangles

commute. Applying M yields the commutativity of the right-hand triangle, and this yields $Mf(\bot) = \bot$, as desired.

Next, for all A the left-hand triangle in the diagram



commutes since it is the special case of the previous one with $f = \eta_A$. The right-hand triangle exhibits one of the unit laws of the monad M. Thus the outside of the diagram commutes. This implies $\mu_A(\bot) = \bot$.

Assume $\bot \in \eta_A[A]$ for some set A, i. e., there exists $a \in A$ with $\eta_A(a) = \bot$. Then $\eta_B(b) = \bot$ for all sets B and all $b \in B$: in fact, let $c_b : A \to B$ be the constant function with value b; by naturality of η and the fact that Mc_b preserves the least element we have

$$\bot = (Mc_b)(\bot) = (Mc_b \cdot \eta_A)(a) = (\eta_B \cdot c_b)(a) = \eta_B(b)\,.$$

From the unit law $\mu_B \cdot \eta_{MB} = \mathrm{id}_{MB}$ it follows $MB \cong 1$ and $\mu_B \cong \mathrm{id}_1$ for all nonempty sets B. For $B = M\emptyset$ (which is nonempty since $\bot \in M\emptyset$) this yields $MM\emptyset \cong 1$, and using the unit law again we conclude $M\emptyset \cong 1$ and $\mu_\emptyset \cong \mathrm{id}_1$. Altogether this means that (M, η, μ) is the trivial monad $(C_1, !, \mathrm{id}_1)$, a contradiction to our assumptions. Thus, finally, $\bot \notin \eta_A[A]$ for all sets A. □

Definition 7.42. Let $e : X \to H_\Sigma X + MA$ be an M-equation morphism. A variable $x \in X$ is called *infinitely unfolding* if there exists an infinite list $x_0, x_1, \ldots$ of variables $x_i \in X$ with $x_0 = x$ and with x_{j+1} appearing in the term $e(x_j) \in H_\Sigma X$ for all $j \in \mathbb{N}$.

Lemma 7.43. *In every $\bar{H}_\Sigma$-algebra every M-equation morphism e has the least solution s, and this satisfies $s(x) = \bot$ for every infinitely unfolding variable x of e.*

Proof. By continuity of F (cf. Assumption 7.38) it follows from the Kleene fixpoint theorem that F has a least fixed point s, i. e., a solution of e in $a : H_\Sigma A \to MA$, and we have $s = \bigsqcup_{i<\omega} s_i$, where $s_0 = \bot$ and $s_{i+1} = Fs_i$. Since the CPO-structure on $\mathsf{Set}(X, MA)$ is pointwise, we have $s(x) = \bigsqcup_{i<\omega} s_i(x)$ for all $x \in X$. We now prove that $s_i(x) = \bot$ for every infinitely unfolding variable x and every $i < \omega$; then $s(x) = \bot$ follows as desired.

We use induction on i. The base case is obvious. For the induction step we assume that $s_i(y) = \bot$ for all infinitely unfolding variables y. Now let

x be an infinitely unfolding variable. Then $e(x) = \sigma(\dots, y, \dots) \in H_\Sigma X$ for another one, y. We obtain

$$\begin{aligned} s_{i+1}(x) &= (Fs_i)(x) \\ &= ([\mu_A, \mathrm{id}_{MA}] \cdot (Ma + \mathrm{id}_{MA}) \cdot (\lambda_A + \mathrm{id}_{MA}) \cdot (H_\Sigma s_i + \mathrm{id}_{MA}) \cdot e)(x) \\ &= ([\mu_A \cdot Ma \cdot \lambda_A, \mathrm{id}_{MA}] \cdot (H_\Sigma s_i + \mathrm{id}_{MA}))(\sigma(\dots, y, \dots)) \\ &= [\mu_A \cdot Ma \cdot \lambda_A, \mathrm{id}_{MA}](\sigma(\dots, s_i(y), \dots)) \\ &= [\mu_A \cdot Ma \cdot \lambda_A, \mathrm{id}_{MA}](\sigma(\dots, \bot, \dots)) \\ &= (\mu_A \cdot Ma)(\bot) \\ &= \bot \end{aligned}$$

The second equation holds by the definition of F written in Set, the last but one equation holds by (7.10), and the last equation holds since Ma and μ_A (by Lemma 7.41) preserve the least element. □

Theorem 7.44. *Under Assumption 7.38 there exists for every λ-CIA $a : H_\Sigma A \to MA$ a well-founded order $>$ on A such that all operations of $a : H_\Sigma A \to MA$ are strictly increasing in the sense that $a_0 > a_1$ whenever $\eta_A(a_0) \sqsubseteq \sigma^A(\dots, a_1, \dots)$ for some algebra operation σ^A.*

Proof. Let $a : H_\Sigma A \to MA$ be a λ-CIA. From the CPO-structure on MA we construct a well-founded order $>$ on A in two steps:

1. Whenever $\eta_A(a_0) \sqsubseteq \sigma^A(b_1, \dots, b_n)$ for some algebra operation σ^A and some values $a_0, b_1, \dots, b_n \in A$ we let $a_0 > b_i$ for all $1 \le i \le n$.

2. We take the reflexive transitive closure.

We only need to verify that the relation defined in (1) is well-founded, i. e., there exists no infinitely descending chain $a_0 > a_1 > a_2 > \cdots$. Then it follows that $>$ is a well-founded order.

Suppose that we have an infinitely descending chain $a_0 > a_1 > a_2 > \cdots$. This means that we have an infinite list

$$\begin{aligned} \eta_A(a_0) &\sqsubseteq \sigma_0^A(b_{0,1}, \dots, a_1, \dots, b_{0,n_0}) \\ \eta_A(a_1) &\sqsubseteq \sigma_1^A(b_{1,1}, \dots, a_2, \dots, b_{1,n_1}) \\ \eta_A(a_2) &\sqsubseteq \sigma_2^A(b_{2,1}, \dots, a_3, \dots, b_{2,n_2}) \\ &\vdots \end{aligned}$$

From this list we construct an M-equation morphism $e : X \to H_\Sigma X + MA$ (written as a system of equations) as follows:

$$\begin{array}{llll} x_0 = \sigma_0(x_{b_{0,1}}, \ldots, x_1, \ldots, x_{b_{0,n_0}}) & x_{b_{0,1}} = \eta_A(b_{0,1}) & x_{b_{1,1}} = \eta_A(b_{1,1}) & \\ x_1 = \sigma_1(x_{b_{1,1}}, \ldots, x_2, \ldots, x_{b_{1,n_1}}) & \vdots & \vdots & \cdots \\ x_2 = \sigma_2(x_{b_{2,1}}, \ldots, x_3, \ldots, x_{b_{2,n_2}}) & x_{b_{0,n_0}} = \eta_A(b_{0,n_0}) & x_{b_{1,n_1}} = \eta_A(b_{1,n_1}) & \\ \quad \vdots & & & \end{array}$$

Observe that in this system the variable x_0 is infinitely unfolding (note that one may have $b_{i,j} = b_{i',j'}$ where $i \neq i'$ or $j \neq j'$; in this case we have $x_{b_{i,j}} = x_{b_{i',j'}}$). We obtain a solution s' of e in $a : H_\Sigma A \to MA$ as follows: we define a function $s'_0 : X \to MA$ by $s'_0(x_i) = \eta_A(a_i)$ for every $i \in \mathbb{N}$ and by $s'_0(x_{b_{i,j}}) = \eta_A(b_{i,j})$ for all other variables. We have $s'_0 \sqsubseteq F s'_0$ since for every $i \in \mathbb{N}$ we have

$$\begin{aligned} (F s'_0)(x_i) &= (\mu_A \cdot Ma \cdot \lambda_A)(\sigma_i(s'_0(x_{b_{i,1}}), \ldots, s'_0(x_{i+1}), \ldots, s'_0(x_{b_{i,n_i}}))) \\ &= (\mu_A \cdot Ma \cdot \lambda_A)(\sigma_i(\eta_A(b_{i,1}), \ldots, \eta_A(a_{i+1}), \ldots, \eta_A(b_{i,n_i}))) \\ &= (\mu_A \cdot Ma \cdot \lambda_A \cdot H_\Sigma \eta_A)(\sigma_i(b_{i,1}, \ldots, a_{i+1}, \ldots, b_{i,n_i})) \\ &= (\mu_A \cdot Ma \cdot \eta_{H_\Sigma A})(\sigma_i(b_{i,1}, \ldots, a_{i+1}, \ldots, b_{i,n_i})) \\ &= \sigma_i^A(b_{i,1}, \ldots, a_{i+1}, \ldots, b_{i,n_i}) \\ &\sqsupseteq \eta_A(a_i) \\ &= s'_0(x_i) \end{aligned}$$

and for all other variables we have $F s'_0(x_{b_{i,j}}) = \eta_A(b_{i,j}) = s'_0(x_{b_{i,j}}) \in MA$. By assumption, F is continuous, hence monotone, thus we have the chain $s'_0 \sqsubseteq F s'_0 \sqsubseteq F^2 s'_0 \sqsubseteq \cdots$ and can define s' to be its join. Then s' is a fixed point for F, and consequently s' is a solution of e in $a : H_\Sigma A \to MA$. Notice that $\eta_A(a_0) \sqsubseteq s'(x_0)$, and since we have $\eta_A(a_0) \neq \bot$ by Lemma 7.41, we see that $s'(x_0) \neq \bot$. By Lemma 7.43 the least solution s of e in $a : H_\Sigma A \to MA$ has $s(x) = \bot$ for every infinitely unfolding variable. So we have $s(x_0) = \bot$. Thus there are two different solutions s and s' of e in $a : H_\Sigma A \to MA$, which is a contradiction to (A, a) being a λ-CIA. So the assumption that there exists an infinitely descending chain is false, and we have proved well-foundedness of the order $>$. □

Characterization of Partial CIAs

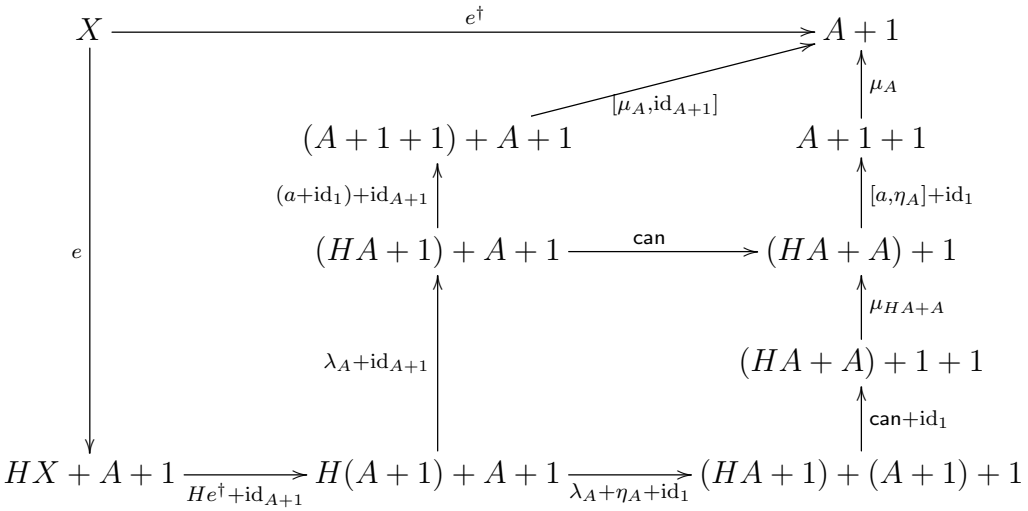
We now turn to the characterization of λ-CIAs for the maybe monad and a polynomial functor H. First we need the following

Lemma 7.45. *Let H be an endofunctor with a distributive law λ over the maybe monad $M = \mathrm{Id} + 1$. Then an $\bar{H}$-algebra is a λ-CIA iff it is a Kleisli-CIA.*

Proof. Let (A, a) be an $\bar{H}$-algebra. Observe that for the maybe monad M we have

$$M(HX + A) = HX + A + 1 = HX + MA\,,$$

and so M-equation morphisms (cf. Definition 7.5) and flat equation morphisms in Set_M (cf. Definition 2.30) coincide. Now it is straightforward to verify that diagram (7.2) commutes for a map $e^\dagger : X \nrightarrow A$ iff diagram (2.1) commutes for $e^\dagger$: consider the diagram



The outside square is diagram (2.1), and the left-hand part is diagram (7.2) (both written in Set). The two right-hand parts are easily seen to commute using $\mu_Y = \mathrm{id}_Y + \nabla_1$ and $\eta_Y = \mathsf{inl} : Y \to Y + 1$, see Example 6.1(1). Thus, $e^\dagger$ is a solution of the M-equation morphism $e : X \to HX + MA$ iff $e^\dagger$ is a solution of (the same) flat equation morphism $e : X \to M(HX + A)$. □

Notation 7.46. Let $a : H_\Sigma A \to A+1$ be an $\bar{H}_\Sigma$-algebra. We extend $\sigma^A : A^n \to A+1$ from Notation 7.8 to $A+1$ strictly, that means that $\sigma^A(s_1, \ldots, s_n) = \bot$ if $s_i = \bot$ for some $1 \le i \le n$ (this is the component of $\mu_A \cdot Ma \cdot \lambda_A : H_\Sigma MA \to MA$ corresponding to $\sigma \in \Sigma_n$, where λ is the canonical distributive law of Example 6.14(1)).

Theorem 7.47. *Let $MX = X + 1$ be the maybe monad, let H_Σ be a finitary polynomial functor, and let $\lambda : H_\Sigma M \to MH_\Sigma$ be the distributive law of Example 6.14(1). Then the following three conditions are equivalent:*

1. *(A, a) is λ-CIA,*

2. *(A, a) is Kleisli-CIA,*

3. *(A, a) is an $\bar{H}_\Sigma$-algebra, and there exists a well-founded order $>$ on A such that every n-ary algebra operation σ^A is strictly increasing in the sense that for all $a_1, \dots, a_n \in A$ with $\sigma^A(a_1, \dots, a_n) \neq \bot$ we have $\sigma^A(a_1, \dots, a_n) > a_i$ for $i = 1, \dots, n$.*

Proof. From Lemma 7.45 we obtain (1) $\Leftrightarrow$ (2). Furthermore, in the current setting Assumption 7.38 is fulfilled with $\sqsubseteq_A$ taken to be the flat CPO-structure, see Remark 7.39. Clearly, the conditions on the orders of Theorems 7.47(3) and 7.44 are the same: in fact, $\eta_A(a_0) \sqsubseteq \sigma^A(\dots, a_1, \dots)$ means $a_0 = \sigma^A(\dots, a_1, \dots)$ here. Thus, we have (1) $\Rightarrow$ (3) by Theorem 7.44.

It remains to prove (3) $\Rightarrow$ (1). So let (A, a) be an $\bar{H}_\Sigma$-algebra satisfying (3), and let $e : X \to H_\Sigma X + MA$ be an M-equation morphism. Next we prove that the least solution s of e in (A, a), which exists by Lemma 7.43, is the only solution of e in (A, a). So suppose we have another solution $s' \neq s$. Since $s \sqsubseteq s'$, there exists a variable x with $s(x) = \bot$ and $s'(x) = a_0$ for some $a_0 \in A$. We must have $e(x) \in H_\Sigma X$, since otherwise diagram (7.2), applied to the solutions s and s', would yield $s(x) = e(x) = s'(x)$. Thus $e(x) = \sigma(x_1, \dots, x_n)$ for some $\sigma \in \Sigma_n$, and hence we obtain from diagram (7.2)

$$\begin{aligned} s(x) &= (\mu_A \cdot (a + \mathrm{id}_1) \cdot \lambda_A \cdot H_\Sigma s)(\sigma(x_1, \dots, x_n)) \\ &= \sigma^A(s(x_1), \dots, s(x_n))\,; \end{aligned}$$

and analogously for s'. Thus we have

$$a_0 = \sigma^A(s'(x_1), \dots, s'(x_n)) \quad \text{and} \quad \bot = \sigma^A(s(x_1), \dots, s(x_n))\,.$$

This implies that there exists a variable x_i, $1 \leq i \leq n$, with $s(x_i) \neq s'(x_i)$. Moreover, $s'(x_i) \neq \bot$ because otherwise we would have

$$a_0 = \sigma^A(s'(x_1), \dots, s'(x_n)) = \sigma^A(\dots, \bot, \dots) = \bot\,.$$

Thus $s'(x_i) = a_1$ for some $a_1 \in A$, and $a_0 = \sigma^A(\dots, a_1, \dots)$ yields $a_0 > a_1$, because the algebra operations are all strictly increasing.

Having $s(x_i) \neq s'(x_i)$, we can repeat the whole argument with x_i in lieu of x; continuing in this way we obtain an infinite sequence $a_0 > a_1 > a_2 > \cdots$, a contradiction to our assumption of well-foundedness of the order $>$. We conclude that s is indeed the only solution of e in (A, a). Thus (A, a) is a λ-CIA. $\square$

Characterization of Nondeterministic CIAs

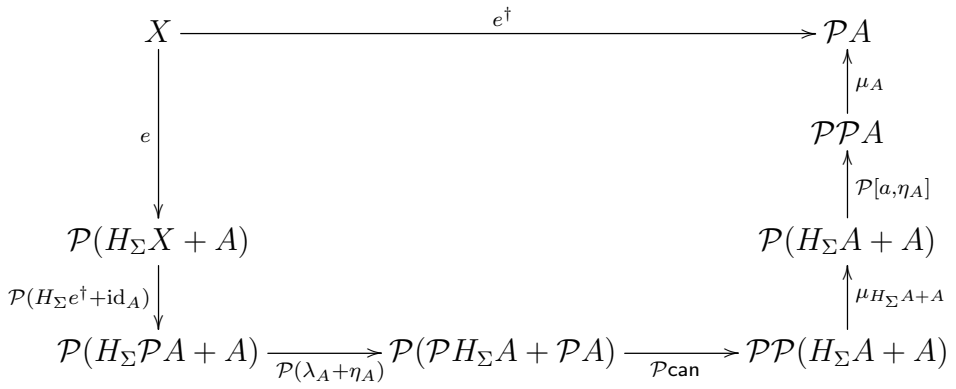
Next we turn to a characterization of λ-CIAs for the powerset monad $M = \mathcal{P}$, a finitary polynomial functor H_Σ and the corresponding canonical distributive law λ.

Notation 7.48. Let $a : H_\Sigma A \to \mathcal{P}A$ be an $\bar{H}_\Sigma$-algebra. We extend $\sigma^A : A^n \to \mathcal{P}A$ from Notation 7.8 to subsets of A as usual: for subsets $A_1, \ldots, A_n \subseteq A$ we write

$$\sigma^A(A_1, \ldots, A_n) = \bigcup \{\sigma^A(a_1, \ldots, a_n) \mid a_i \in A_i \text{ for all } i = 1, \ldots, n\}\,.$$

Notice that for each $\sigma \in \Sigma_n$ the extended σ^A is the component of $\mu_A \cdot Ma \cdot \lambda_A : H_\Sigma MA \to MA$ corresponding to σ.

Remark 7.49. Observe that for a flat equation morphism $e : X \nrightarrow \bar{H}_\Sigma X + A$, diagram (2.1) yields the following commutative diagram for a solution $e^\dagger : X \nrightarrow A$ in the $\bar{H}_\Sigma$-algebra (A, a):



The commutativity of this diagram is equivalent to the following equation for every variable $x \in X$:

$$e^\dagger(x) = (e(x) \cap A) \cup \bigcup_{\substack{\sigma(x_1,\ldots,x_n) \\ \in (e(x) \cap H_\Sigma X)}} \sigma^A(e^\dagger(x_1), \ldots, e^\dagger(x_n)) \tag{7.12}$$

Theorem 7.50. *Let $M = \mathcal{P}$ be the powerset monad, let H_Σ be a finitary polynomial functor, and let $\lambda : H_\Sigma M \to MH_\Sigma$ be the distributive law of Example 6.14(2). Then the following three conditions are equivalent:*

1. *(A, a) is λ-CIA,*
2. *(A, a) is Kleisli-CIA,*

3. *(A, a) is an $\bar{H}_\Sigma$-algebra, and there exists a well-founded order $>$ on A such that every n-ary algebra operation σ^A is strictly increasing in the sense that for all $a_1, \dots, a_n \in A$ with $\sigma^A(a_1, \dots, a_n) = B$ we have $b > a_i$ for all $b \in B$ and $i = 1, \dots, n$.*

Proof. We have (2) $\Rightarrow$ (1) by Proposition 7.12. Furthermore, in the current setting Assumption 7.38 is fulfilled with $\sqsubseteq_A$ taken to be set inclusion, see Remark 7.39. Clearly, the conditions on the orders of Theorems 7.50(3) and 7.44 are the same: in fact, $\eta_A(a_0) \sqsubseteq \sigma^A(\dots, a_1, \dots)$ means $a_0 \in \sigma^A(\dots, a_1, \dots)$ here. Thus, we have (1) $\Rightarrow$ (3) by Theorem 7.44.

It remains to prove (3) $\Rightarrow$ (2). So let $a : \bar{H}_\Sigma A \nrightarrow A$ be an $\bar{H}_\Sigma$-algebra satisfying (3), and let $e : X \nrightarrow \bar{H}_\Sigma X + A$ be a flat equation morphism (in $\mathsf{Set}_\mathcal{P}$). Analogously to the map F of (7.11) we obtain from diagram (2.1) a map

$$G : f \mapsto [a, \mathrm{id}_A] \cdot (\bar{H}_\Sigma f + \mathrm{id}_A) \cdot e$$

on $\mathsf{Set}(X, \mathcal{P}A)$, which is clearly continuous. Let $s : X \to \mathcal{P}A$ be the least fixed point of G, i. e., s is the least solution of e in A. We shall prove that s is the only solution of e in A. So suppose we have another solution with $s' \neq s$. Since $s \sqsubseteq s'$, there exists a variable x and an element $a_0 \in A$ such that $a_0 \notin s(x)$ but $a_0 \in s'(x)$. From equation (7.12) we obtain

$$s(x) = (e(x) \cap A) \cup \bigcup_{\substack{\sigma(x_1,\dots,x_n) \\ \in (e(x) \cap H_\Sigma X)}} \sigma^A(s(x_1), \dots, s(x_n))$$

for s, and analogously for s'. Thus,

$$a_0 \in \bigcup_{\substack{\sigma(x_1,\dots,x_n) \\ \in (e(x) \cap H_\Sigma X)}} \sigma^A(s'(x_1), \dots, s'(x_n))$$

and

$$a_0 \notin \bigcup_{\substack{\sigma(x_1,\dots,x_n) \\ \in (e(x) \cap H_\Sigma X)}} \sigma^A(s(x_1), \dots, s(x_n)) \,.$$

From these equations we know that there must be some $\sigma(x_1, \dots, x_n) \in e(x) \cap H_\Sigma X$ with $a_0 \in \sigma^A(s'(x_1), \dots, s'(x_n))$ but $a_0 \notin \sigma^A(s(x_1), \dots, s(x_n))$. Consequently, we have some variable x_i, $1 \leq i \leq n$, such that there exists an element $a_1 \in A$ with $a_1 \in s'(x_i)$ and $a_1 \notin s(x_i)$ which causes $a_0 \in \sigma^A(s'(x_1), \dots, s'(x_n))$, i. e. $a_0 \in \sigma^A(\dots, a_1, \dots)$. From the latter we conclude $a_0 > a_1$, because the algebra operations are all strictly increasing.

We can repeat the argument for a_1 and x_i in lieu of a_0 and x and obtain an infinite sequence $a_0 > a_1 > a_2 > \cdots$. This is a contradiction to our

assumption of well-foundedness of the order $>$. Thus A is a Kleisli-CIA, which completes the proof. □

Corollary 7.51. *For $M = \mathcal{P}$, an $\overline{\mathrm{Id}}$-algebra is a λ-CIA (or, equivalently, a Kleisli-CIA) iff the corresponding graph (cf. Example 7.11(3)) is well-founded.*

In fact, if a graph is well-founded, then the induced (strict) order is; and conversely, a subgraph (corresponding to the $\overline{\mathrm{Id}}$-algebra) of a well-founded graph (constructed from a well-founded order) is well-founded.

Notice that this result also shows that even though Kleisli- and λ-CIAs coincide, and free λ-CIAs (and Kleisli-CIAs) coincide with free $\bar{H}$-algebras, not every $\bar{H}$-algebra needs to be a λ-CIA; in fact, every directed graph yields an $\overline{\mathrm{Id}}$-algebra, and clearly there are non-well-founded such graphs.

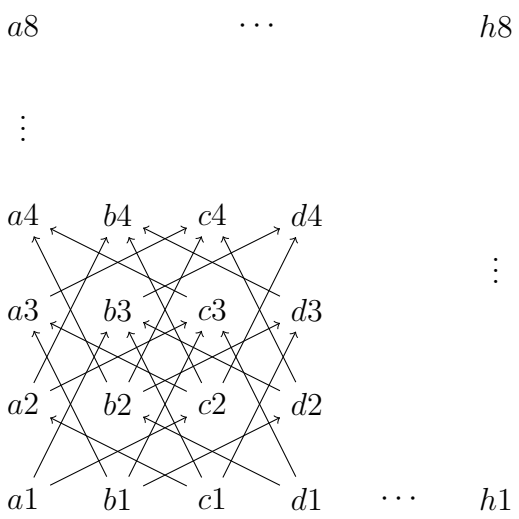
Examples 7.52. 1. Let $A = \{a_0, a_1, a_2\}$, and define a binary operation $*$ by

$$\begin{array}{lll} a_0 * a_0 = \{a_1, a_2\} & a_1 * a_0 = \emptyset & a_2 * a_0 = \emptyset \\ a_0 * a_1 = \{a_2\} & a_1 * a_1 = \{a_2\} & a_2 * a_1 = \emptyset \\ a_0 * a_2 = \emptyset & a_1 * a_2 = \emptyset & a_2 * a_2 = \emptyset \end{array}$$

Here we have the well-founded order $a_2 > a_1 > a_0$ for which the operation $*$ is strictly increasing, thus A is a λ-CIA and a Kleisli-CIA by Theorem 7.50.

2. Let $A = \{2, 3, \ldots, 100\} \subset \mathbb{N}$ be the $\bar{H}$-algebra of Example 7.4. For the usual order on A the operations are clearly strictly increasing, and well-foundedness follows from finiteness of A, i. e., A satisfies condition (3) in Theorem 7.50 and hence is a λ-CIA and a Kleisli-CIA. Observe that for $\mathcal{P}$-equation morphisms $e : X \to HX + \mathcal{P}A$ an *infinitely unfolding variable* x, i. e., one for which there exists an infinite sequence $x_0, x_1, x_2, \ldots$ with $x_0 = x$ and each x_{i+1} appearing in a flat term $e(x_i) \in HX$, is always solved to $\emptyset$ in a λ-CIA. However, for flat equation morphisms $e' : X \nrightarrow HX + A$ infinitely unfolding variables may be solved to nonempty sets, see Example 7.4.

3. Consider the nondeterministic algebra (A, next^+) from the introduction to Part II. The corresponding graph



is well-founded since its paths have length 7 at most. By Corollary 7.51, (A, next^+) is a Kleisli-CIA, i. e. the equation morphism e corresponding to the equation $x = \{b1, \mathsf{op}(x)\}$ from the introduction has indeed the unique solution $e^\dagger$ in the nondeterministic algebra (A, next^+) given by $e^\dagger(x) = K$, where K is the set displayed in Figure II.2.

Characterization of Composite CIAs

We turn to a characterization of λ-CIAs for the environment monad and for arbitrary endofunctors H on Set. Here $\lambda : HM \to MH$ is the distributive law of Example 6.23(2). Recall from Example 6.1(5) that for the unit of $M = (-)^E$ we have $\pi_i^X \cdot \eta_X = \mathrm{id}_X$ for every $i \in E$ and that the multiplication μ_X makes the following squares commute for every $i \in E$:

$$\begin{array}{ccc} (X^E)^E & \xrightarrow{\mu_X} & X^E \\ {\scriptstyle \pi_i^{X^E}}\downarrow & & \downarrow{\scriptstyle \pi_i^X} \\ X^E & \xrightarrow{\pi_i^X} & X \end{array}$$

Furthermore recall from Example 6.23(2) that $\pi_i^{HX} \cdot \lambda_X = H\pi_i^X$ for every $i \in E$.

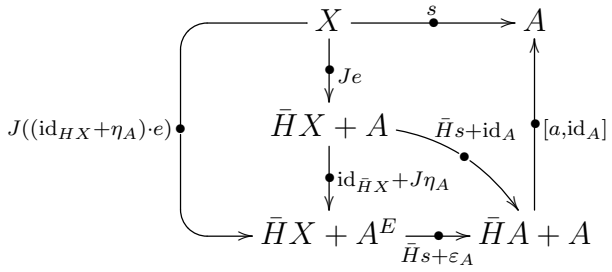
Theorem 7.53. *Let $MX = X^E$ be the environment monad, let H be an endofunctor on Set, and let $\lambda : HM \to MH$ be the distributive law of Example 6.23(2). Then the following three conditions are equivalent:*

1. *(A, a) is a λ-CIA,*
2. *(A, a) is a Kleisli-CIA,*

3. *(A, a) is an $\bar{H}$-algebra such that for every $i \in E$ the H-algebra $(A, \pi_i \cdot a)$ is a CIA in* Set.

Proof. The implication (2) $\Rightarrow$ (1) is settled by Proposition 7.12.

We prove (1) $\Rightarrow$ (3): let $a : HA \to A^E$ be a λ-CIA. We are going to show that each $a_i = \pi_i \cdot a : HA \to A$ is a CIA for H. Let $e : X \to HX + A$ be a flat equation morphism. Form the M-equation morphism $(\mathrm{id}_{HX} + \eta_A) \cdot e : X \to HX + A^E$ and take its unique solution $s : X \to A^E$. Then the outside of the diagram below commutes:

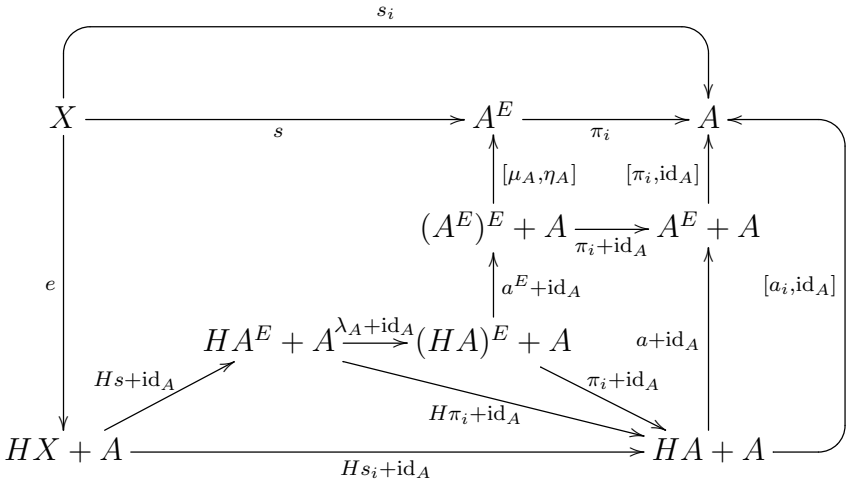


(7.13)

Observe that the left-hand part commutes by the fact that J preserves coproducts and composition, and the lower triangle commutes by the adjunction law $\varepsilon_{JA} \cdot J\eta_A = \mathrm{id}$. Thus the upper right-hand part commutes. We define for each $i \in E$

$$s_i = (X \xrightarrow{s} A^E \xrightarrow{\pi_i} A)$$

and prove that s_i is a solution of e in (A, a_i) for all $i \in E$. To this end consider the following diagram:



(7.14)

The big inner part is the upper right-hand part of (7.13) written in Set, so it commutes. All other inner parts are also easily seen to commute; use the

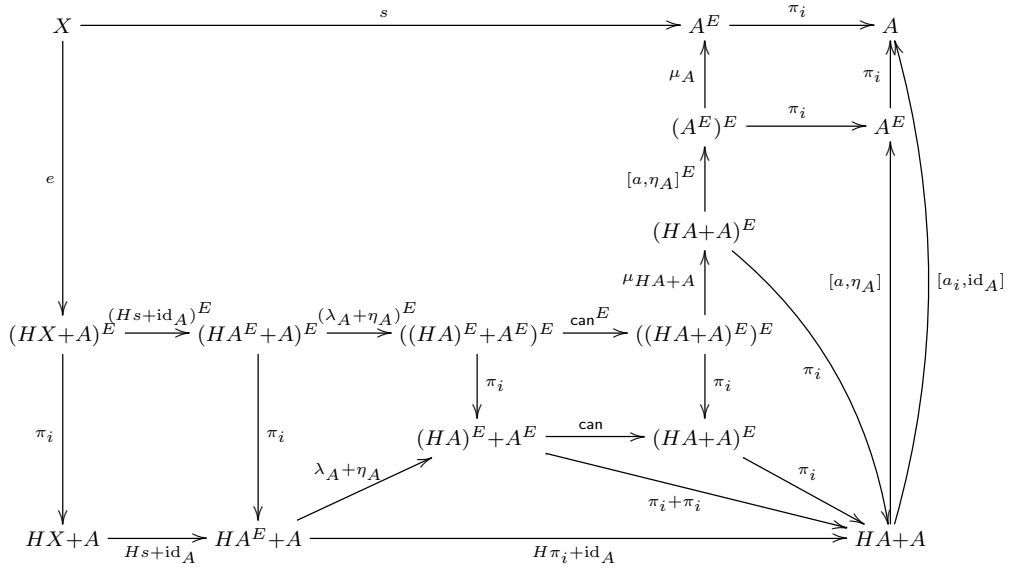
definitions of μ, η, λ, a_i and s_i and naturality of π_i. Thus the outside of (7.14) commutes, showing s_i to be a solution of e in (A, a_i).

To prove the uniqueness, suppose that s'_j is one particular solution of e in (A, a_j) for some $j \in E$. Define a morphism $s : X \to A^E$ by

$$\pi_j \cdot s = s'_j \quad \text{and} \quad \pi_i \cdot s = s_i \quad \text{for all } i \neq j\,.$$

Then for each $i \in E$ the outside of diagram (7.14) commutes (with s_j substituted by s'_j). Hence, the big inner part commutes which is the upper right-hand part of diagram (7.13). Thus the outside of diagram (7.13) commutes, too, and so s is the unique solution of $(\mathrm{id}_{HX} + \eta_A) \cdot e$ in the λ-CIA (A, a). Thus, we have $s'_j = \pi_j \cdot s = s_j$, as desired.

Finally, we prove (3) $\Rightarrow$ (2): let each $a_i = \pi_i \cdot a : HA \to A$ be a CIA. We are going to show that $a : HA \to A^E$ is a Kleisli-CIA. Let $e : X \to\!\!\!\!\bullet\, HX + A$ be a flat equation morphism (in Set_M), i. e., $e : X \to (HX + A)^E$. Then for every $i \in E$ we have the flat equation morphism $\pi_i \cdot e : X \to HX + A$ which has the unique solution $s_i : X \to A$ in the CIA (A, a_i). We define s by $\pi_i \cdot s = s_i$ for all $i \in E$ and prove that s is a unique solution of e in (A, a). We first show that $[a, \mathrm{id}_A] \cdot (\bar{H}s + \mathrm{id}_A) \cdot e = s$ in Set_M. This means, we show that the big upper left-hand square in the diagram below commutes:



The outside commutes for each $i \in E$ since $s_i = \pi_i \cdot s$ solves $\pi_i \cdot e$, and all the other inner parts also commute (by the definitions of μ, η, λ and a_i and the naturality of π_i). Thus the desired big inner square commutes when composed with any π_i, hence it commutes.

If s is any solution of e in (A, a), then the upper left-hand square commutes, whence the outside commutes for each $i \in E$. This means that $\pi_i \cdot s$

is a solution of $\pi_i \cdot e$, thus we have $s_i = \pi_i \cdot s$ by the uniqueness of solutions in (A, a_i). This shows the uniqueness of s and thus completes the proof. $\square$

7.4 Related Work

Since Kleisli CIAs are ordinary CIAs in a Kleisli category and λ-CIAs are a related concept, our approach and results are sometimes related to those from Milius' paper [Mil05] about CIAs. This applies to the choice of morphisms between λ-CIAs (see Proposition 7.15) and the characterization of unary λ-CIAs (see Examples 7.11).

Our most general result on free λ-CIAs and Kleisli-CIAs (Corollary 7.21) is based on the work [HJS07] of Hasuo, Jacobs and Sokolova. The proof of Proposition 7.36 dualizes a proof from the paper [AMV06a] by Adámek, Milius and Velebil.

Most of the contents of Chapter 7 were published in the joint paper [MPS09] with Milius and Palm. However, the full proofs were not included in that conference paper and are added here. Also the parts of Section 7.2 dealing with the nonempty powerset monad and canonical liftings (which can be found in our paper [Sch11]) were not in the paper [MPS09]. Finally, we added to Remark 7.9 and added Example 7.52(3).

Chapter 8

Recursive Program Schemes and Effects

In Chapter 7 we were concerned with systems of equations with effects which were solved in algebras with effects. We found that for the maybe monad, the powerset monad and the environment monad Kleisli-CIAs and λ-CIAs are reasonable notions of algebras with effects in which systems of equations have unique solutions. We also saw in Section 7.2 that the subdistribution monad can be handled analogously to the maybe and powerset monads, and that the environment monad specializes to the identity monad in case $E = 1$, which leads back to the simpler case without effects. However, Kleisli-CIAs or λ-CIAs for the nonempty powerset monad were missing—we only obtained the weaker result of Proposition 7.36, for which we needed techniques specific for nondeterminism (see e. g. Definitions 7.30 and 7.31) in addition to the methods from our category theoretic framework.

The present chapter confirms this exceptional position of the nonempty powerset monad on the level of recursive program schemes: we shall establish a framework of techniques and a notion of a recursive program scheme with effects which has, provided that it is guarded, unique uninterpreted solutions in case of all monads from Example 6.1 except the nonempty powerset monad. However, here we make the effort to show that this monad yet fits into our framework if this is completed by techniques specific for nondeterminism: analyzing the effect of "nonempty nondeterminism" we shall find that there are no unique, but canonical solutions.

In this chapter partial, nondeterministic and probabilistic RPS's ($M = \mathrm{Id} + 1$, $M = \mathcal{P}$, $M = \mathcal{D}$) are treated together; nonempty nondeterministic ($M = \mathcal{P}^+$) and compositional ($M = (-)^E$ which includes $M = \mathrm{Id}$ as the special case $E = 1$) RPS's are dealt with separately. The chapter is structured as follows: in Section 8.1 we see how distributive laws of (algebra) functors

H over monads M (cf. Section 6.2) can be extended to distributive laws of free monads F^H or sometimes even free CIMs T^H over monads M. These distributive laws induce composite monads which are shown to be weak CIMs for $M = \mathcal{P}^+$ and CIMs for $M = (-)^E$ in Section 8.2. A central technical result is reached in Section 8.3 where certain final coalgebras (or weakly final coalgebras for $M = \mathcal{P}^+$) are exhibited. In Section 8.4 special morphisms into these (weakly) final coalgebras are proved to be monad morphisms. This prepares us to prove our main results in Section 8.5, namely that every guarded RPS with one of the effects under consideration has a unique uninterpreted solution—except for the effect of "nonempty nondeterminism", where we obtain canonical uninterpreted solutions.

8.1 Extending to Distributive Laws of Monads

In Section 6.2 we have seen that distributive laws of functors H over monads M allow to apply an algebra to results of effectful computations as in (6.1). In the present chapter, where we investigate uninterpreted solutions of recursive program schemes, we are not concerned with the application of algebras, but with purely syntactic manipulations. We have argued above (the last item below (6.1)) that distributive laws $\lambda : HM \to MH$ are appropriate also for this purpose due to their naturality. In fact, they rewrite operation symbols applied to effect-structured arguments to effect-structured operation symbols applied to plain arguments.

We shall also need such a rewriting mechanism which deals with arbitrary terms instead of operation symbols (which are flat terms). Clearly (at least for finite terms) this can be achieved by repeated application of λ. More elegantly, we use again distributive laws, this time of the free monad F on H over the monad M. Indeed, recall that for a polynomial endofunctor H on Set the free monad (F, η^H, μ^H) gives the sets FX of (finite) terms over variables from X built up from operation symbols according to H, where $\eta^H_X : X \to FX$ considers variables as terms of depth 0 and $\mu^H_X : FFX \to FX$ flattens terms with terms as variables to a single term. Then a distributive law $\lambda' : FM \to MF$ of monads is precisely the right notion: it rewrites terms of operation symbols applied to effect-structured variables to effect-structured terms applied to plain variables. In addition to the unit and multiplication laws for M which we already explained for λ above (see below (6.1)), we have unit and multiplication laws for F:

- in the case of a term of depth 0 (i. e. an effect-structured variable) appli-

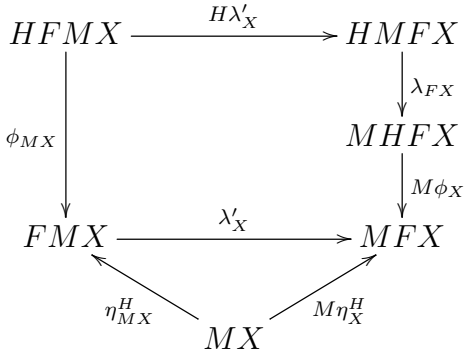
cation of the distributive law just considers this as an effect-structured term of depth 0: this follows from the unit law $\lambda'_X \cdot \eta^H_{MX} = M\eta^H_X$;

- due to the multiplication law $\lambda'_X \cdot \mu^H_{MX} = M\mu^H_X \cdot \lambda'_{FX} \cdot F\lambda'_X$, it does not matter whether the effect-structure is drawn outside a term at once or step by step for subterms.

Extending to Finite Terms

Although we explained distributive laws $\lambda' : FM \to MF$ only for polynomial functors H, we shall not require H to be polynomial but work with arbitrary functors H for which free algebras exist. Recall from Theorem 2.22 that in this case the free monad (F, η^H, μ^H) on H exists; also recall the natural transformations ϕ and κ from this theorem.

Definition 8.1. Given a distributive law $\lambda : HM \to MH$ of a functor H over the monad M, we define for every set X the morphism $\lambda'_X : FMX \to MFX$ to be the unique homomorphism to the H-algebra $M\phi_X \cdot \lambda_{FX}$ extending $M\eta^H_X$. That is, λ'_X is uniquely determined by the following commutative diagram:



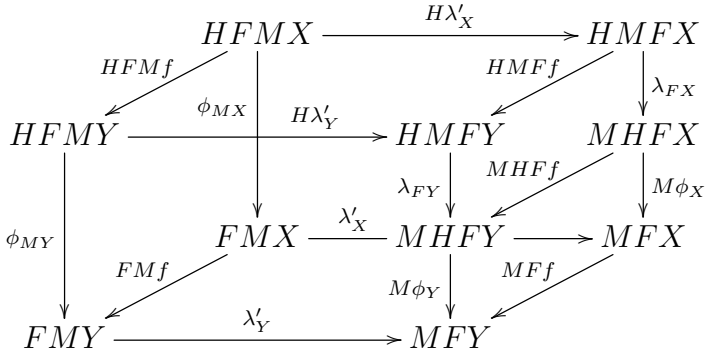
(8.1)

Proposition 8.2. *The morphisms λ'_X from Definition 8.1 constitute a distributive law $\lambda' : FM \to MF$ of monads.*

Proof. To prove the proposition we need to verify naturality of λ' and the four laws for a distributive law of monads.

(1) The morphisms λ'_X form a natural transformation λ'. To prove this,

let $f : X \to Y$ be an arbitrary morphism and consider the following diagram:



Both paths of the bottom square are homomorphisms between H-algebras as we can see from the commuting "walls": the left-hand wall commutes by naturality of ϕ, the right-hand wall by naturality of λ and ϕ, and the remaining walls by the definition of λ'_X and λ'_Y. Furthermore both paths are the unique homomorphism extending $M(\eta^H_Y \cdot f)$ as we check in

$$MFf \cdot \lambda'_X \cdot \eta^H_{MX} = MFf \cdot M\eta^H_X = M(\eta^H_Y \cdot f)$$

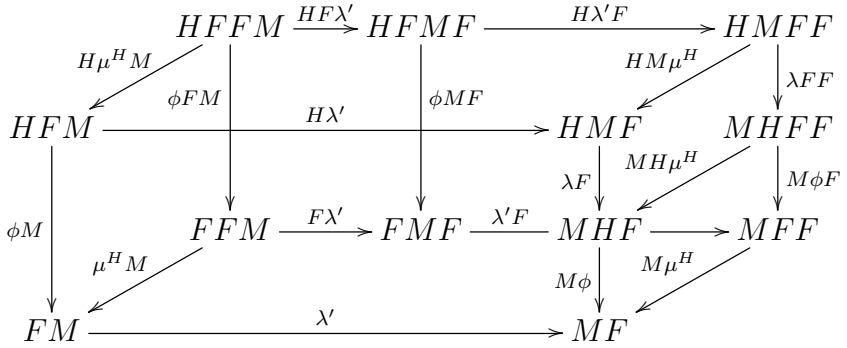
and

$$\lambda'_Y \cdot FMf \cdot \eta^H_{MX} = \lambda'_Y \cdot \eta^H_{MY} \cdot Mf = M\eta^H_Y \cdot Mf = M(\eta^H_Y \cdot f)\,.$$

Thus the desired bottom square commutes.

(2) The unit law involving η^H holds by the definition of λ', see the lower part of (8.1).

(3) The multiplication law involving μ^H holds. Consider the following diagram:



We need to show that the bottom square commutes. To this end we consider it componentwise and see that for every set X both paths in the bottom square are the unique homomorphism to the H-algebra $M\phi_X \cdot \lambda_{FX}$:

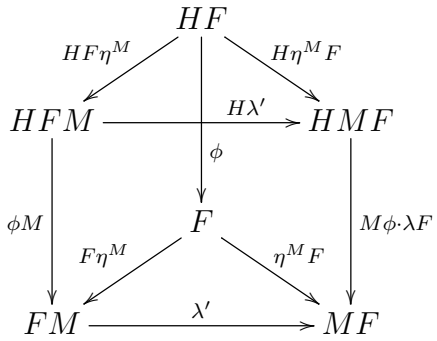
$HMFX \to MFX$ extending λ'_X: the homomorphism property is established for each component by the commuting "walls": commutativity of the left-hand wall and the lower part of the right-hand wall is due to Remark 2.23, the upper part of the right-hand wall is due to naturality of λ, and the remaining walls are due to naturality of ϕ and the definition of λ'. Furthermore we calculate

$$\lambda'_X \cdot \mu^H_{MX} \cdot \eta^H_{FMX} = \lambda'_X$$

and

$$\begin{aligned} M\mu^H_X \cdot \lambda'_{FX} \cdot F\lambda'_X \cdot \eta^H_{FMX} &= M\mu^H_X \cdot \lambda'_{FX} \cdot \eta^H_{MFX} \cdot \lambda'_X \\ &= M\mu^H_X \cdot M\eta^H_{FX} \cdot \lambda'_X \\ &= \lambda'_X \,. \end{aligned}$$

(4) The unit law involving η^M holds. Consider the following diagram:

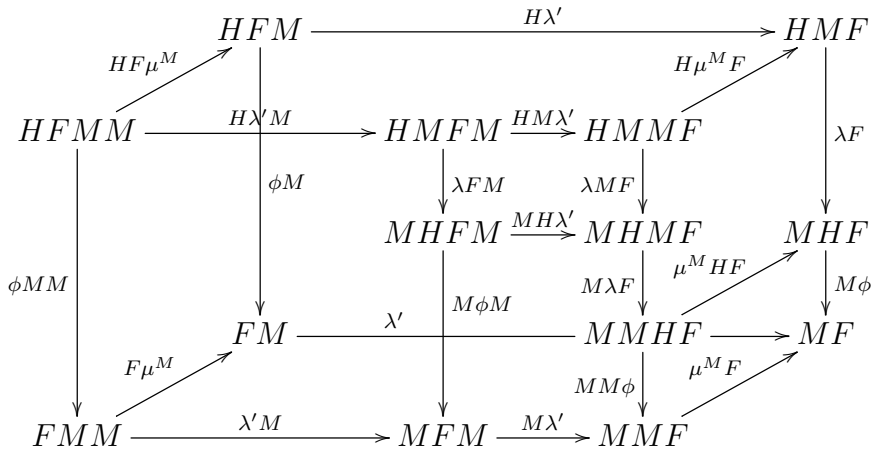


We need to show that the bottom triangle commutes. To this end we consider it componentwise and see that for every set X both paths in the bottom triangle are the unique homomorphism to the H-algebra $M\phi_X \cdot \lambda_{FX} : HMFX \to MFX$ extending $\eta^M_{FX} \cdot \eta^H_X$: η^M_{FX} is a homomorphism by the unit law for λ and naturality of η^M, and it trivially extends $\eta^M_{FX} \cdot \eta^H_X$. The other path is a homomorphism by naturality of ϕ and by the definition of λ', and we calculate

$$\lambda'_X \cdot F\eta^M_X \cdot \eta^H_X = \lambda'_X \cdot \eta^H_{MX} \cdot \eta^M_X = M\eta^H_X \cdot \eta^M_X = \eta^M_{FX} \cdot \eta^H_X \,.$$

(5) The multiplication law involving μ^M holds. Consider the following

diagram:



We need to show that the bottom square commutes. To this end we consider it componentwise and see that for every set X both paths in the bottom square are the unique homomorphism to the H-algebra $M\phi_X \cdot \lambda_{FX}$ extending $M\eta^H_X \cdot \mu^M_X$: the homomorphism property is established by the commuting "walls" of the diagram. The left-hand wall commutes due to naturality of ϕ, the right-hand wall due to the multiplication law for λ and naturality of μ^M, and for the remaining walls use the definition of λ' and naturality of λ. Furthermore we calculate

$$\lambda'_X \cdot F\mu^M_X \cdot \eta^H_{MMX} = \lambda'_X \cdot \eta^H_{MX} \cdot \mu^M_X = M\eta^H_X \cdot \mu^M_X$$

and

$$\begin{aligned} \mu^M_{FX} \cdot M\lambda'_X \cdot \lambda'_{MX} \cdot \eta^H_{MMX} &= \mu^M_{FX} \cdot M\lambda'_X \cdot M\eta^H_{MX} \\ &= \mu^M_{FX} \cdot MM\eta^H_X \\ &= M\eta^H_X \cdot \mu^M_X \,. \end{aligned}$$

□

We did not make any assumption except for the existence of free algebras for the functor H. Thus most of the distributive laws we have seen in Section 6.2 can be extended this way; in particular, we can apply Proposition 8.2 to all canonical distributive laws from Theorem 6.19.

Examples 8.3. We describe the extension $\lambda' : FM \to MF$ of the canonical distributive laws $\lambda : HM \to MH$ from Examples 6.14.

1. For the maybe monad $M = \mathrm{Id} + 1$, λ'_X maps a finite term over variables from $X + \{\bot\}$ to itself if $\bot$ does not appear in the term; otherwise, the term is mapped to $\bot$.

2. For the powerset monad $M = \mathcal{P}$, λ'_X maps a finite term over "variable sets" from $\mathcal{P}X$ to the set of all terms over X which have the same shape as the original term, but where from each variable set one variable is chosen.

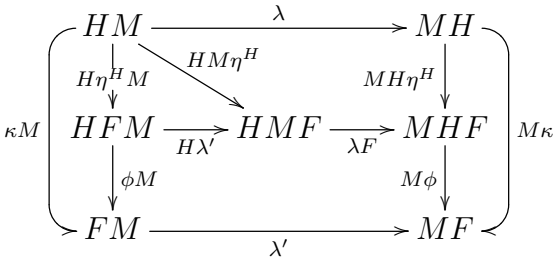
3. For the subdistribution monad $M = \mathcal{D}$, λ'_X maps a finite term over subdistributions on X to the subdistribution on terms over X which assigns to each term with the same shape as the original term the product of the probabilities of all the variables used in the term; to all other terms it assigns 0.

4. For the nonempty powerset monad $M = \mathcal{P}^+$, λ'_X is defined as for the powerset monad above, but no empty sets occur.

5. For the environment monad $M = (-)^E$, λ'_X maps a finite term over E-indexed families from X^E to the E-indexed family of terms over X where for every $i \in E$ the i-component is given by choosing the i-components in the variables. For the special case $E = 1$ of the identity monad $M = \mathrm{Id}$, λ'_X is just id_{FX}.

Lemma 8.4. *The distributive law λ' of Proposition 8.2 fulfils $\lambda' \cdot \kappa M = M\kappa \cdot \lambda$.*

Proof. In the diagram



all inner parts commute: the left-hand and right-hand parts are the definition of κ (see Theorem 2.22), the upper triangle and the lower square are the definition of λ' in diagram (8.1) and the remaining upper square commutes due to naturality of λ. Thus the outside commutes. □

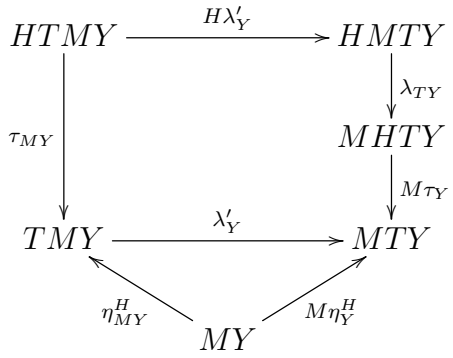
Extending to Infinite Terms: Environment Monad

As we shall see next, for some monads we can even extend distributive laws to infinite terms. Instead of the existence of free H-algebras ϕ_Y we require the existence of final coalgebras for the functors $H(-)+Y$ (or, equivalently, of free

CIAs τ_Y on Y for H), i. e. we require H to be iteratable, see Definition 2.50. Recall from Theorem 2.47 that in this case the free CIM (T, η^H, μ^H) on H exists; also recall the natural transformations τ and κ from this theorem.

We rephrase Definition 8.1 for H-CIAs in place of H-algebras.

Definition 8.5. Let $\lambda : HM \to MH$ be a distributive law of a functor H over the monad M. Let $M\tau_Y \cdot \lambda_{TY} : HMTY \to MTY$ be an H-CIA for every set Y. We define for every set Y the morphism $\lambda'_Y : TMY \to MTY$ to be the unique homomorphism to the H-CIA $M\tau_Y \cdot \lambda_{TY}$ extending $M\eta^H_Y$. That is, λ'_Y is uniquely determined by the following commutative diagram:



(8.2)

Proposition 8.6. *The morphisms λ'_Y from Definition 8.5 constitute a distributive law $\lambda' : TM \to MT$ of monads.*

Proof. Since the homomorphisms between CIAs simply are H-algebra homomorphisms, the proof is the same as for Proposition 8.2 with the free monad (F, η^H, μ^H) replaced by the free CIM (T, η^H, μ^H) and with ϕ replaced by τ. Instead of Remark 2.23 use Remark 2.48, and instead of diagram (8.1) use diagram (8.2) with $M = (-)^E$, of course. □

Corollary 8.7. *The canonical distributive law λ of an iteratable functor H over the environment monad $M = (-)^E$ from Example 6.23(2) induces a distributive law $\lambda' : TM \to MT$ of monads.*

Proof. This follows from Proposition 8.6 applied to $M = (-)^E$ since $M\tau_Y \cdot \lambda_{TY}$ is an H-CIA for every set Y by Corollary 7.25 and Remark 7.6. □

Observation 8.8. For the special case of $E = 1$, we have $M = \mathrm{Id}$ and we have for any iteratable endofunctor H the canonical distributive law $\lambda = \mathrm{id}$ over M, see Example 6.23(2). Definition 8.5 yields the identity maps $\lambda'_Y = \mathrm{id}_{HY}$ and thus trivially the distributive law $\lambda' = \mathrm{id}$ of the monad T over the monad Id.

Remark 8.9. Lemma 8.4 still holds for the distributive laws obtained from Proposition 8.6: just replace (F, η^H, μ^H) by (T, η^H, μ^H) and ϕ by τ. Instead of Theorem 2.22 see Theorem 2.47 for the definition of κ, and instead of diagram (8.1) use diagram (8.2) with $M = (-)^E$, of course.

Extending to Infinite Terms: Nonempty Powerset Monad

Here we require H to be a finitary polynomial functor. In particular this implies that a set TY consists of rooted, ordered, finitely branching trees which may have leaves labeled in Y, see Theorem 2.47 together with Example 2.29.

On the sets $\mathcal{P}^+Y$ we have the partial order $\subseteq$. This induces a partial order $\sqsubseteq$ on $T\mathcal{P}^+Y$ as follows:

$$t_1 \sqsubseteq t_2 \quad \Longleftrightarrow \quad \begin{array}{l} t_1 \text{ and } t_2 \text{ have the same shape and for every leaf labeled} \\ \text{by a set } Y_1 \in \mathcal{P}^+Y \text{ in } t_1 \text{ and by a set } Y_2 \in \mathcal{P}^+Y \text{ in } t_2 \text{ we} \\ \text{have } Y_1 \subseteq Y_2. \end{array}$$

The partial order $\sqsubseteq$ in turn lifts pointwise to a partial order on $\mathsf{Set}(X, T\mathcal{P}^+Y)$ for every set X which we again denote by $\sqsubseteq$, i. e. $f \sqsubseteq g \Leftrightarrow \forall x \in X : f(x) \sqsubseteq g(x)$.

Recall from Proposition 7.36 that for canonical distributive laws λ of a finitary polynomial set functor H over $\mathcal{P}^+$, the algebra $\mathcal{P}^+\tau_Y \cdot \lambda_{TY}$ is an H-CEA for every set Y. Since τ_Y is not only the free H-CIA, but also the free H-CEA (see Theorem 2.40 and Corollary 2.35), we can apply Definition 8.5 to the nonempty powerset monad $M = \mathcal{P}^+$ to obtain unique homomorphisms λ'_Y between CEAs for every set Y.

We are now going to prove that the λ'_Y constitute a distributive law $\lambda' : T\mathcal{P}^+ \to \mathcal{P}^+T$ of monads (see Proposition 8.12 below). But different from Propositions 8.2 and 8.6, we need to work with homomorphisms between CEAs instead of algebras or CIAs which are not just algebra homomorphisms, but solution preserving algebra homomorphisms (see Definition 2.38). In order to prove some of the morphisms relevant for a proof analogous to Propositions 8.2 and 8.6 to be homomorphisms between CEAs, we shall use Lemma 8.11 below. However, for other relevant morphisms it seems difficult to prove that they are homomorphisms between CEAs. But we are still able to prove the remaining axioms for λ' by hand, i. e. by arguing how the involved morphisms act. To this end, we prove a description of the action of λ'_Y in the following

Lemma 8.10. *The maps $\lambda'_Y : T\mathcal{P}^+Y \to \mathcal{P}^+TY$ from Definition 8.5 (applied to the canonical distributive law λ of a finitary polynomial functor H over the monad $M = \mathcal{P}^+$ from Example 6.14(4)) act as follows: given a tree*

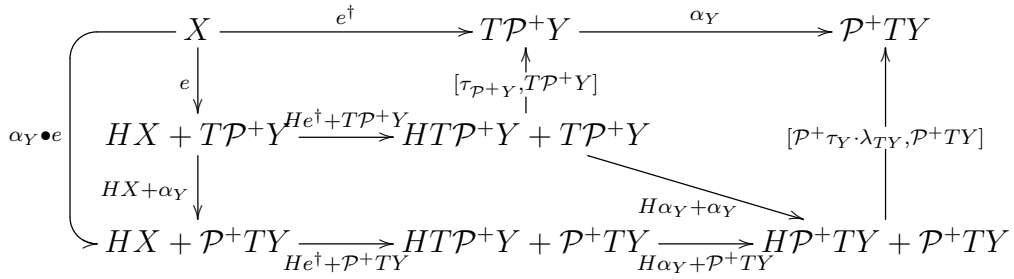
$t \in T\mathcal{P}^+Y$ where leaves may be labeled by nonempty subsets of Y, $\lambda'_Y(t)$ is the set of all trees obtained by choosing in each of these leaves one element from the labeling set.

Proof. Let us denote the maps described in the statement of the lemma by $\alpha_Y : T\mathcal{P}^+Y \to \mathcal{P}^+TY$. To show $\alpha_Y = \lambda'_Y$ for every set Y it suffices to prove for every set Y that α_Y is the unique homomorphism between the free CEA $\tau_{\mathcal{P}^+Y}$ and the CEA $\mathcal{P}^+\tau_Y \cdot \lambda_{TY}$ from Proposition 7.36 extending $\mathcal{P}^+\eta^H_Y$.

It is easy to check that α_Y extends $\mathcal{P}^+\eta^H_Y$, i. e. that we have $\alpha_Y \cdot \eta^H_{\mathcal{P}^+Y} = \mathcal{P}^+\eta^H_Y$: to consider a nonempty subset of Y as a label of a singleton tree and then taking the set of all (singleton) trees labeled by an element from this set obviously is the same as to make every element of the nonempty set into a singleton tree labeled by this element.

In order to show that α_Y is a homomorphism of CEAs it suffices to prove that it is solution preserving; from this it follow that α_Y is an H-algebra homomorphism as shown in [AMV06a], Lemma 5.2. Let $e : X \to HX + T\mathcal{P}^+Y$ be a flat equation morphism with parameters in $T\mathcal{P}^+Y$. We need to prove $\alpha_Y \cdot e^\dagger = (\alpha_Y \bullet e)^\dagger$; since the right-hand side is the greatest solution of $\alpha_Y \bullet e$ in the CEA $\mathcal{P}^+\tau_Y \cdot \lambda_{TY}$ by Proposition 7.33, it suffices to prove that the left-hand side also is a greatest solution in this CEA.

(1) $\alpha_Y \cdot e^\dagger$ is a solution. Consider the following diagram:



The left-hand part commutes by the definition of $\bullet$; the upper left-hand square commutes since $e^\dagger$ is the (unique) solution of e in the free CEA $\tau_{\mathcal{P}^+Y}$. The lower part is trivial, and the remaining right-hand part commutes since α_Y is a homomorphism between the H-algebras $\tau_{\mathcal{P}^+Y}$ and $\mathcal{P}^+\tau_Y \cdot \lambda_{TY}$. In fact, the equation $\mathcal{P}^+\tau_Y \cdot \lambda_{TY} \cdot H\alpha_Y = \alpha_Y \cdot \tau_{\mathcal{P}^+Y}$ holds since it is the same for an element from $HT\mathcal{P}^+Y$ to take all choices of leaf labels in the child trees and then taking all choices at the root as to take all choices immediately. Thus, the outside commutes showing $\alpha_Y \cdot e^\dagger$ to be a solution of $\alpha_Y \bullet e$.

(2) $\alpha_Y \cdot e^\dagger$ is a greatest solution. We prove $\alpha_Y \cdot e^\dagger \sqsupseteq (\alpha_Y \bullet e)^\dagger$ elementwise, i. e. we prove $(\alpha_Y \cdot e^\dagger)(x) \supseteq (\alpha_Y \bullet e)^\dagger(x)$ for every $x \in X$. Let $t \in (\alpha_Y \bullet e)^\dagger(x)$, i. e. we have $\overline{\alpha_Y \bullet e} : \bar{X} \to H\bar{X} + TY$ over $\alpha_Y \bullet e$ via $f : \bar{X} \to X$ such that

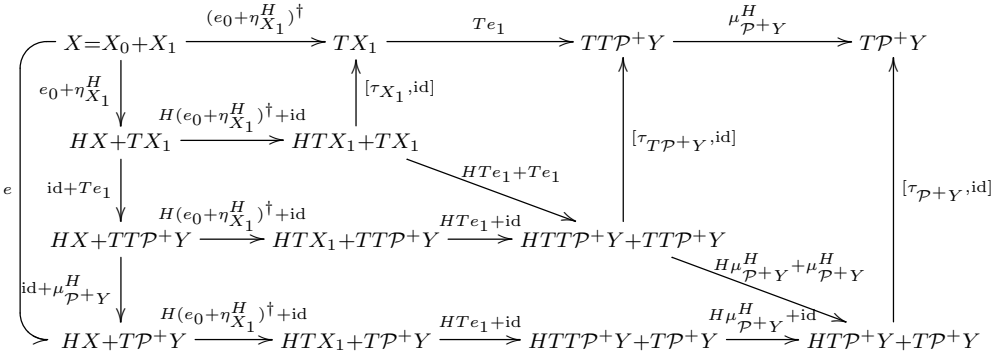
$f(\bar{x}) = x$ and $t = (\overline{\alpha_Y \bullet e})^\dagger(\bar{x})$. More concrete, splitting $X = X_0 + X_1$ with $e[X_0] \subseteq HX$ and $e[X_1] \subseteq T\mathcal{P}^+Y$, splitting $\bar{X} = \bar{X}_0 + \bar{X}_1$ with $f[\bar{X}_0] \subseteq X_0$ and $f[\bar{X}_1] \subseteq X_1$ and splitting morphisms with domain X or $\bar{X}$ accordingly, this means

$$(Hf \cdot \overline{\alpha_Y \bullet e}_0)(\bar{x}) = ((\alpha_Y \bullet e)_0 \cdot f_0)(\bar{x}) \quad \text{for all } \bar{x} \in \bar{X}_0$$

and

$$\overline{\alpha_Y \bullet e}_1(\bar{x}) \in ((\alpha_Y \bullet e)_1 \cdot f_1)(\bar{x}) \quad \text{for all } \bar{x} \in \bar{X}_1\,.$$

Consider the following diagram:



The left-hand part commutes by one of the monad unit laws for T; the upper left-hand square commutes since $(e_0+\eta^H_{X_1})^\dagger$ is the (unique) solution of $e_0+\eta^H_{X_1}$ in τ_{X_1}; the two lower left-hand parts trivially commute and for the two upper right-hand squares use naturality of τ and Remark 2.48. Thus all inner parts commute, and so does the outside which shows that $e^\dagger = \mu^H_{\mathcal{P}^+Y} \cdot Te_1 \cdot (e_0+\eta^H_{X_1})^\dagger$ for the unique solution of e in $\tau_{\mathcal{P}^+Y}$.

We have a similar diagram for the solution of $T\eta^+_Y \bullet \overline{\alpha_Y \bullet e}$ showing that $(T\eta^+_Y \bullet \overline{\alpha_Y \bullet e})^\dagger = \mu^H_{\mathcal{P}^+Y} \cdot T(T\eta^+_Y \cdot \overline{\alpha_Y \bullet e}_1) \cdot (\overline{\alpha_Y \bullet e}_0 + \eta^H_{\bar{X}_1})^\dagger$.

Recall from right before this lemma that for every set Z we have a partial order $\sqsubseteq$ on $\mathsf{Set}(Z, T\mathcal{P}^+Y)$. Let $d : V \to TW$ be a function and define the function $\psi : \mathsf{Set}(W, T\mathcal{P}^+Y) \to \mathsf{Set}(V, T\mathcal{P}^+Y)$ by $\psi(g) = \mu^H_{\mathcal{P}^+Y} \cdot Tg \cdot d$ for every $g : W \to T\mathcal{P}^+Y$. It is not difficult to see that ψ is monotone: $g \sqsubseteq h$ means $g(w) \sqsubseteq h(w)$ for every $w \in W$ which means that the trees $g(w)$ and $h(w)$ have the same shape and the sets labeling leaves of $g(w)$ are subsets of the sets labeling the corresponding leaves of $h(w)$. For every $v \in V$, we need to prove that $\psi(g)(v) = (\mu^H_{\mathcal{P}^+Y} \cdot Tg \cdot d)(v) \sqsubseteq (\mu^H_{\mathcal{P}^+Y} \cdot Th \cdot d)(v) = \psi(h)(v)$. But both sides just substitute trees that are related by $\sqsubseteq$ for variables of the tree $d(v)$ and consider the results as trees again; thus these trees have the same shape and the sets in corresponding leaves are related by $\subseteq$, so the trees are related by $\sqsubseteq$.

We apply this monotonicity result to $V = \bar{X}$, $W = \bar{X}_1$, $d = (\overline{\alpha_Y \bullet e}_0 + \eta^H_{\bar{X}_1})^\dagger : \bar{X} \to T\bar{X}_1$, $g = T\eta^+_Y \cdot \overline{\alpha_Y \bullet e}_1$ and $h = e_1 \cdot f_1$, where $T\eta^+_Y \cdot \overline{\alpha_Y \bullet e}_1 \sqsubseteq e_1 \cdot f_1$ follows from the above result $\overline{\alpha_Y \bullet e}_1(\bar{x}) \in ((\alpha_Y \bullet e)_1 \cdot f_1)(\bar{x}) = (\alpha_Y \cdot e_1 \cdot f_1)(\bar{x})$ for all $\bar{x} \in \bar{X}_1$ and the concrete description of α_Y. Then we see that the inequality

$$\begin{aligned}
T\eta^+_Y \cdot (\overline{\alpha_Y \bullet e})^\dagger &\overset{(*)}{=} (T\eta^+_Y \bullet \overline{\alpha_Y \bullet e})^\dagger \\
&= \mu^H_{\mathcal{P}^+Y} \cdot T(T\eta^+_Y \cdot \overline{\alpha_Y \bullet e}_1) \cdot (\overline{\alpha_Y \bullet e}_0 + \eta^H_{\bar{X}_1})^\dagger \\
&\sqsubseteq \mu^H_{\mathcal{P}^+Y} \cdot T(e_1 \cdot f_1) \cdot (\overline{\alpha_Y \bullet e}_0 + \eta^H_{\bar{X}_1})^\dagger \\
&= \mu^H_{\mathcal{P}^+Y} \cdot Te_1 \cdot Tf_1 \cdot (\overline{\alpha_Y \bullet e}_0 + \eta^H_{\bar{X}_1})^\dagger \\
&\overset{(*)}{=} \mu^H_{\mathcal{P}^+Y} \cdot Te_1 \cdot (\overline{\alpha_Y \bullet e}_0 + Tf_1 \cdot \eta^H_{\bar{X}_1})^\dagger \\
&\overset{(\star)}{=} \mu^H_{\mathcal{P}^+Y} \cdot Te_1 \cdot (e_0 + \eta^H_{X_1})^\dagger \cdot f \\
&= e^\dagger \cdot f
\end{aligned}$$

holds where the equations marked by $(*)$ are true since $T\eta^+_Y$ and Tf_1 are H-algebra homomorphisms between CIAs and thus are solution preserving, see [Mil05], Proposition 2.3; and the equation marked by $(\star)$ is proved by the commutative diagram

$$\begin{array}{ccccc}
\bar{X} & \xrightarrow{\quad f \quad} & X & \xrightarrow{\quad (e_0+\eta^H_{X_1})^\dagger \quad} & TX_1 \\
\Big\downarrow{\scriptstyle \overline{\alpha_Y \bullet e}_0 + Tf_1 \cdot \eta^H_{\bar{X}_1}} & & \Big\downarrow{\scriptstyle e_0+\eta^H_{X_1}} & & \Big\uparrow{\scriptstyle [\tau_{X_1}, \mathrm{id}]} \\
H\bar{X} + TX_1 & \xrightarrow[Hf+\mathrm{id}]{} & HX + TX_1 & \xrightarrow[H(e_0+\eta^H_{X_1})^\dagger+\mathrm{id}]{} & HTX_1 + TX_1
\end{array}$$

where the left-hand square is due to $(Hf \cdot \overline{\alpha_Y \bullet e}_0)(\bar{x}) = ((\alpha_Y \bullet e)_0 \cdot f_0)(\bar{x}) = (e_0 \cdot f_0)(\bar{x})$ for all $\bar{x} \in \bar{X}_0$ and the right-hand square commutes since $(e_0 + \eta^H_{X_1})^\dagger$ is the (unique) solution of $e_0 + \eta^H_{X_1}$ in τ_{X_1}. Thus the outside commutes proving $(\overline{\alpha_Y \bullet e}_0 + Tf_1 \cdot \eta^H_{\bar{X}_1})^\dagger = (e_0 + \eta^H_{X_1})^\dagger \cdot f$ by the uniqueness of solutions in τ_{X_1}.

Applying the above inequality to $\bar{x}$, we obtain

$$T\eta^+_Y(t) = (T\eta^+_Y \cdot (\overline{\alpha_Y \bullet e})^\dagger)(\bar{x}) \sqsubseteq (e^\dagger \cdot f)(\bar{x}) = e^\dagger(x)$$

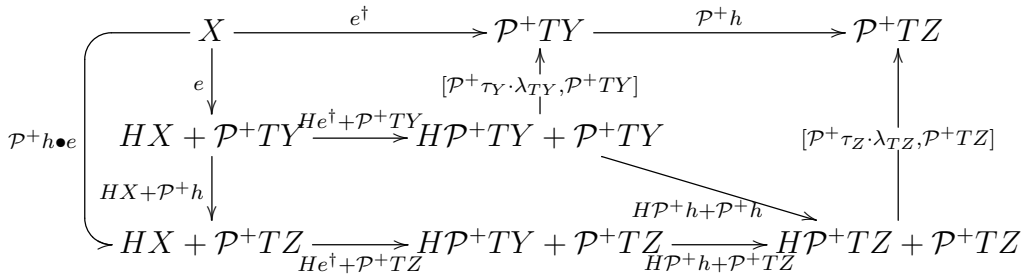
which implies the desired result $t \in (\alpha_Y \cdot e^\dagger)(x)$. □

For the following lemma, recall from before Proposition 7.33 that the sets $\mathsf{Set}(X, \mathcal{P}^+V)$ carry a partial order $\leq$.

Lemma 8.11. *Let $h : TY \to TZ$ be a homomorphism between the H-algebras $\tau_Y : HTY \to TY$ and $\tau_Z : HTZ \to TZ$. Then $\mathcal{P}^+h$ is a homomorphism between the CEAs $\mathcal{P}^+\tau_Y \cdot \lambda_{TY} : H\mathcal{P}^+TY \to \mathcal{P}^+TY$ and $\mathcal{P}^+\tau_Z \cdot \lambda_{TZ} : H\mathcal{P}^+TZ \to \mathcal{P}^+TZ$ from Proposition 7.36.*

Proof. Given any flat equation morphism $e : X \to HX + \mathcal{P}^+TY$ and the H-algebra homomorphism h as in the statement, we must prove that $\mathcal{P}^+h$ preserves solutions, i. e. that $\mathcal{P}^+h \cdot e^\dagger = (\mathcal{P}^+h \bullet e)^\dagger$. We already know that the right-hand side is the greatest solution of $\mathcal{P}^+h \bullet e$, thus it suffices to show that the left-hand side is a solution of $\mathcal{P}^+h \bullet e$ with $\mathcal{P}^+h \cdot e^\dagger \geq (\mathcal{P}^+h \bullet e)^\dagger$.

(1) $\mathcal{P}^+h \cdot e^\dagger$ is a solution. Consider the following diagram:



The left-hand part is the definition of $\bullet$, the upper left-hand square commutes since $e^\dagger$ is a solution of e; the lower part trivially commutes, and for the right-hand part use naturality of λ and that h is an H-algebra homomorphism. Thus the outside commutes showing that $\mathcal{P}^+h \cdot e^\dagger$ is a solution of $\mathcal{P}^+h \bullet e$.

(2) We prove $\mathcal{P}^+h \cdot e^\dagger \geq (\mathcal{P}^+h \bullet e)^\dagger$. We do so by showing $(\mathcal{P}^+h \cdot e^\dagger)(x) \supseteq (\mathcal{P}^+h \bullet e)^\dagger(x)$ for all $x \in X$. To show this, let $t \in (\mathcal{P}^+h \bullet e)^\dagger(x)$. This means that we have $\overline{\mathcal{P}^+h \bullet e} : \bar{X} \to H\bar{X} + TZ$ over $\mathcal{P}^+h \bullet e$ via $f^{\mathcal{P}^+h \bullet e} : \bar{X} \to X$ and some $\bar{x} \in \bar{X}$ with $f^{\mathcal{P}^+h \bullet e}(\bar{x}) = x$ and $(\overline{\mathcal{P}^+h \bullet e})^\dagger(\bar{x}) = t$. We make this property more explicit by splitting $X = X_0 + X_1$ where $e[X_0] \subseteq HX$ and $e[X_1] \subseteq \mathcal{P}^+TY$ and splitting $\bar{X} = \bar{X}_0 + \bar{X}_1$ where $f^{\mathcal{P}^+h \bullet e}[\bar{X}_0] \subseteq X_0$ and $f^{\mathcal{P}^+h \bullet e}[\bar{X}_1] \subseteq X_1$. Accordingly we split morphisms with domain X or $\bar{X}$ into coproducts. Then the fact that $\overline{\mathcal{P}^+h \bullet e}$ is over $\mathcal{P}^+h \bullet e$ via $f^{\mathcal{P}^+h \bullet e}$ can be expressed in the two formulas

$$(Hf^{\mathcal{P}^+h \bullet e} \cdot (\overline{\mathcal{P}^+h \bullet e})_0)(\bar{x}) = ((\mathcal{P}^+h \bullet e)_0 \cdot f_0^{\mathcal{P}^+h \bullet e})(\bar{x}) \quad \text{for all } \bar{x} \in \bar{X}_0$$

and

$$((\overline{\mathcal{P}^+h \bullet e})_1)(\bar{x}) \in ((\mathcal{P}^+h \bullet e)_1 \cdot f_1^{\mathcal{P}^+h \bullet e})(\bar{x}) \quad \text{for all } \bar{x} \in \bar{X}_1\,.$$

We need to show $t \in (\mathcal{P}^+h \cdot e^\dagger)(x)$. To this end we define $f^e : \bar{X} \to X$ by $f^e = f^{\mathcal{P}^+h \bullet e}$ and $\bar{e} : \bar{X} \to H\bar{X} + TY$ by

$$\bar{e}_0(\bar{x}) = (\overline{\mathcal{P}^+h \bullet e})_0(\bar{x}) \quad \text{for all } \bar{x} \in \bar{X}_0$$

and

$$\bar{e}_1(\bar{x}) = s \in (e_1 \cdot f_1^e)(\bar{x}) \text{ such that } (\overline{\mathcal{P}^+h \bullet e})_1(\bar{x}) = h(s) \quad \text{for all } \bar{x} \in \bar{X}_1\,.$$

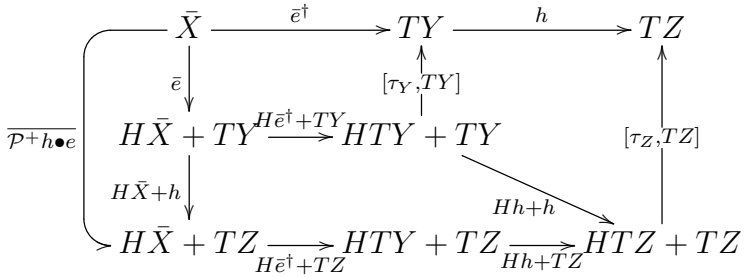
Such trees s always exist since $(\overline{\mathcal{P}^+h \bullet e})_1(\bar{x}) \in ((\mathcal{P}^+h \bullet e)_1 \cdot f_1^{\mathcal{P}^+h \bullet e})(\bar{x}) = (\mathcal{P}^+h \cdot e_1 \cdot f_1^e)(\bar{x})$ for all $\bar{x} \in \bar{X}_1$, so $\bar{e}$ is well-defined. We show that $\bar{e}$ is over e via f^e: clearly we have

$$(Hf^e \cdot \bar{e}_0)(\bar{x}) = (Hf^{\mathcal{P}^+h \bullet e} \cdot (\overline{\mathcal{P}^+h \bullet e})_0)(\bar{x}) = ((\mathcal{P}^+h \bullet e)_0 \cdot f_0^{\mathcal{P}^+h \bullet e})(\bar{x}) = (e_0 \cdot f_0^e)(\bar{x})$$

for all $\bar{x} \in \bar{X}_0$; we also have

$$\bar{e}_1(\bar{x}) = s \in (e_1 \cdot f_1^e)(\bar{x})$$

for all $\bar{x} \in \bar{X}_1$ by definition. Now consider the following diagram:



The left-hand part commutes by the definition of $\bar{e}$; the upper left-hand square commutes since $\bar{e}^\dagger$ is the (unique) solution of $\bar{e}$ in the free CEA τ_Y; the lower part trivially commutes, and for the right-hand part use that h is an H-algebra homomorphism. Thus the outside commutes showing that $h \cdot \bar{e}^\dagger$ is a solution of $\overline{\mathcal{P}^+h \bullet e}$. By the uniqueness of such solutions we have $h \cdot \bar{e}^\dagger = (\overline{\mathcal{P}^+h \bullet e})^\dagger$.
To finish our proof, we see that $t = (\overline{\mathcal{P}^+h \bullet e})^\dagger(\bar{x}) = (h \cdot \bar{e}^\dagger)(\bar{x})$. This gives the desired result since from the fact that $\bar{e}$ is over e it follows $t = (h \cdot \bar{e}^\dagger)(\bar{x}) \in (\mathcal{P}^+h \cdot e^\dagger)(x)$. □

Proposition 8.12. *The canonical distributive law $\lambda : H\mathcal{P}^+ \to \mathcal{P}^+H$ of a finitary polynomial set functor H over the monad $\mathcal{P}^+$ extends to a distributive law $\lambda' : T\mathcal{P}^+ \to \mathcal{P}^+T$ of monads.*

Proof. We prove that the maps λ'_Y from Definition 8.5 (applied to the canonical distributive law λ of a finitary polynomial set functor H over the monad $M = \mathcal{P}^+$ as explained above Lemma 8.10) form a distributive law of monads. In large parts (items (1)–(3) below) the proof is similar to the one of Proposition 8.2; we only replace F by T and ϕ by τ and additionally prove that all homomorphisms involved are solution preserving using Lemma 8.11. In items (4)–(5) below we provide direct proofs of two laws for λ' using Lemma 8.10.

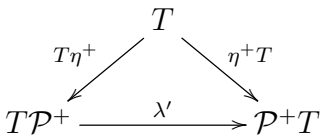
(1) The maps λ'_Y form a natural transformation λ'. Here we need to check that $T\mathcal{P}^+f$ and $\mathcal{P}^+Tf$ are solution preserving for every map $f : X \to Y$ (for

λ'_X and λ'_Y this is immediate from their definition). For $T\mathcal{P}^+ f$ this follows from [Mil05], Proposition 2.3, since this is a homomorphism between CIAs. For $\mathcal{P}^+ Tf$, apply Lemma 8.11 to the H-algebra homomorphism Tf.

(2) The unit law for λ' involving η^H holds. This is immediate from the definition of λ'.

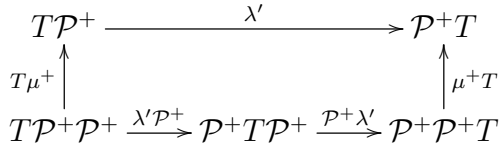
(3) The multiplication law for λ' involving μ^H holds. Here we need to check that $T\lambda'$, $\mu^H\mathcal{P}^+$ and $\mathcal{P}^+\mu^H$ are componentwise solution preserving (for λ' and $\lambda' T$ this is immediate from their definition). For $T\lambda'$ this follows since its components are homomorphisms between CIAs; the same is true for $\mu^H\mathcal{P}^+$, see Remark 2.48. For $\mathcal{P}^+\mu^H$, apply Lemma 8.11 to the components of μ^H which are H-algebra homomorphisms (again by Remark 2.48).

(4) The unit law for λ' involving η^+ holds. We use Lemma 8.10 and Example 6.1(4) to verify that the desired diagram



commutes. From those we see that the diagram just states that to make the leaf labels of a tree singleton sets and then taking the set of all trees where for every such leaf an element of the set is chosen as a label results in the singleton set containing the original tree, which is obviously right.

(5) The multiplication law for λ' involving μ^+ holds. We use Lemma 8.10 and Example 6.1(4) to verify that the desired diagram



commutes. To this end, consider its Y-component and let $s \in T\mathcal{P}^+\mathcal{P}^+ Y$ and

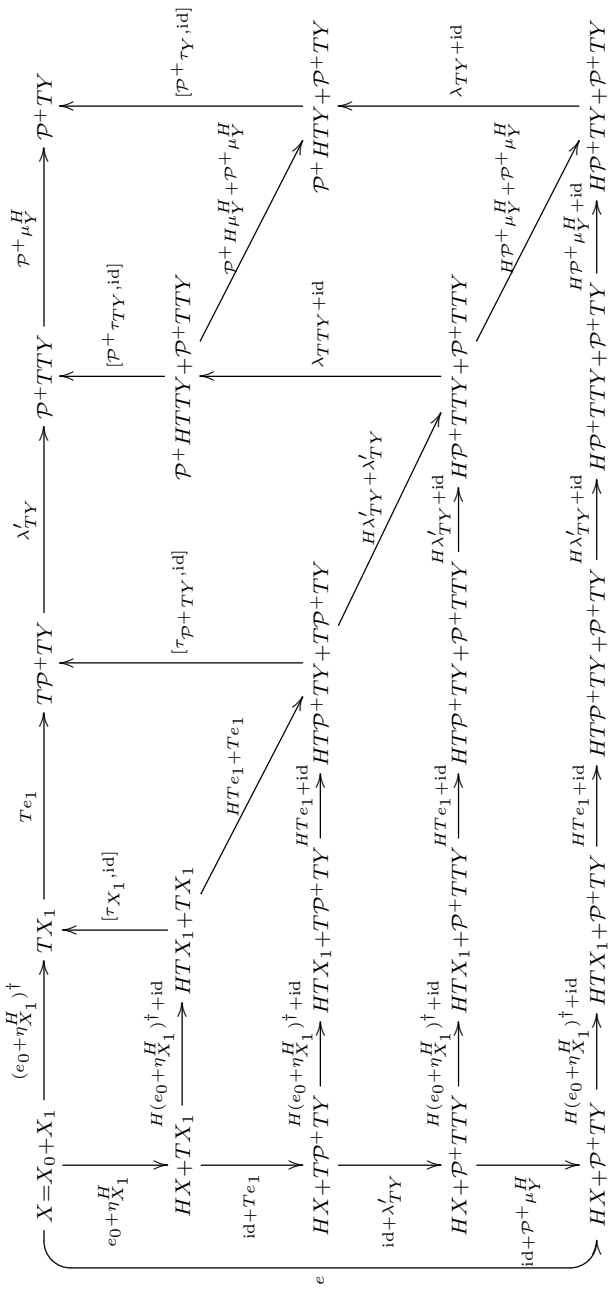
$t \in TY$ be trees. It holds

$$
\begin{array}{ll}
 & t \in (\mu^+ TY \cdot \mathcal{P}^+\lambda'_Y \cdot \lambda'_{\mathcal{P}^+Y})(s) \\
\iff & t \text{ lies in a set that lies in } (\mathcal{P}^+\lambda'_Y \cdot \lambda'_{\mathcal{P}^+Y})(s) \\
\iff & \text{there is a tree with the same shape as } t \text{ in } \lambda'_{\mathcal{P}^+Y}(s) \text{ such that every set by which a leaf is labeled contains the corresponding leaf label of } t \\
\iff & s \text{ has the same shape as } t, \text{ and for every leaf of } t \text{ its label lies in a set that lies in the set by which the corresponding leaf of } s \text{ is labeled} \\
\iff & \text{for every leaf of } t \text{ its label lies in the set by which the corresponding leaf of } T\mu^+(s) \text{ is labeled} \\
\iff & t \in (\lambda'_Y \cdot T\mu^+)(s)
\end{array}
$$

which proves the desired law. □

Remark 8.13. Lemma 8.4 also holds for the distributive laws obtained from Proposition 8.12: just replace (F, η^H, μ^H) by (T, η^H, μ^H) and ϕ by τ. Instead of Theorem 2.22 see Theorem 2.47 for the definition of κ, and instead of diagram (8.1) use diagram (8.2) with $M = \mathcal{P}^+$, of course.

Remark 8.14. From Lemma 8.11 and Proposition 8.12 we obtain a third characterization of the operation $(-)^\dagger$ from Definition 7.31 (besides its definition and the second characterization given in Corollary 7.35): any flat equation morphism $e : X \to HX + \mathcal{P}^+TY$ can be decomposed into $e_0 : X_0 \to HX$ and $e_1 : X_1 \to \mathcal{P}^+TY$ where $X = X_0 + X_1$. We form the flat equation morphism $e_0 + \eta^H_{X_1} : X \to HX + TX_1$ which has a unique solution $(e_0 + \eta^H_{X_1})^\dagger$ in the free CIA τ_{X_1} meaning that the upper left-hand square in the following diagram commutes:



The left-hand part commutes by one of the unit laws for the distributive law λ' and one of the unit laws for the monad T; the three lower left-hand parts commute trivially; and for the four parts in the upper right-hand area use naturality of τ, the upper square of diagram (8.2) with $M = \mathcal{P}^+$, and naturality of λ and Remark 2.48. Since all inner parts commutes so does the outside showing $\mathcal{P}^+\mu^H_Y \cdot \lambda'_{TY} \cdot Te_1 \cdot (e_0 + \eta^H_{X_1})^\dagger$ to be a solution of e.

Moreover, this is the greatest solution: $Te_1 \cdot (e_0 + \eta^H_{X_1})^\dagger$ is a unique solution, λ'_{TY} preserves greatest solutions by its definition and $\mathcal{P}^+\mu^H_Y$ preserves them by Lemma 8.11 since μ^H_Y is an H-algebra homomorphism between τ_{TY} and τ_Y by Remark 2.48. Thus we have $e^\dagger = \mathcal{P}^+\mu^H_Y \cdot \lambda'_{TY} \cdot Te_1 \cdot (e_0 + \eta^H_{X_1})^\dagger$ meaning that taking the greatest solution can be done by unfolding the recursive definition given by e first without the parameters and then plugging the parameter sets in the leaves of the resulting tree and taking the set of trees of all combinations of leaves, cf. the description of λ' in the statement of Lemma 8.10.

8.2 Composite Monads

We start this section with the basic fact that distributive laws of monads give rise to composite monads:

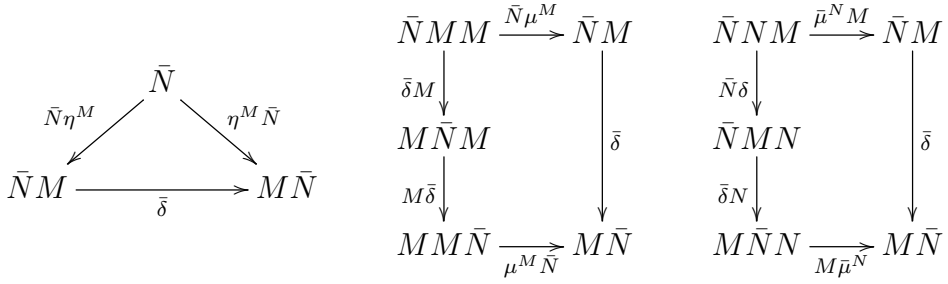
Lemma 8.15 ([Bec69])**.** *Given a distributive law $\delta : NM \to MN$ of monads, $(MN, \eta^M * \eta^N, (\mu^M * \mu^N) \cdot M\delta N)$ is again a (composite) monad.*

Using the extensions (Proposition 8.2) of the canonical distributive laws from Theorem 6.19, we see that for every analytic functor H and every commutative monad M on Set we canonically obtain composite monads MF, where F is the free monad on H. In particular this applies to the maybe monad $M = \mathrm{Id} + 1$, the powerset monad $M = \mathcal{P}$ and the subdistribution monad $M = \mathcal{D}$.

For the environment monad $M = (-)^E$ and the nonempty powerset monad $M = \mathcal{P}^+$, we canonically obtain composite monads MT, where T is the free CIM on H. This follows from Corollary 8.7 and Proposition 8.12. For $M = (-)^E$ this even holds for arbitrary iteratable set functors H, for $M = \mathcal{P}^+$ we are restricted to finitary polynomial set functors H. But in both cases the composite monads have more structure as we prove next. We start with results which are common for both cases. Recall from Definition 2.41 the notion of a module for a monad.

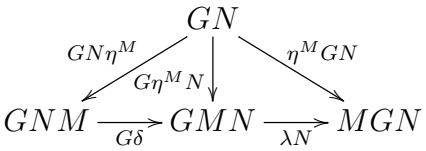
Definition 8.16. Given a distributive law $\delta : NM \to MN$ of monads, by a *δ-distributive law* of an N-module $(\bar{N}, \bar{\mu}^N)$ over the monad M is meant a natural transformation $\bar{\delta} : \bar{N}M \to M\bar{N}$ making the following diagrams

commute:

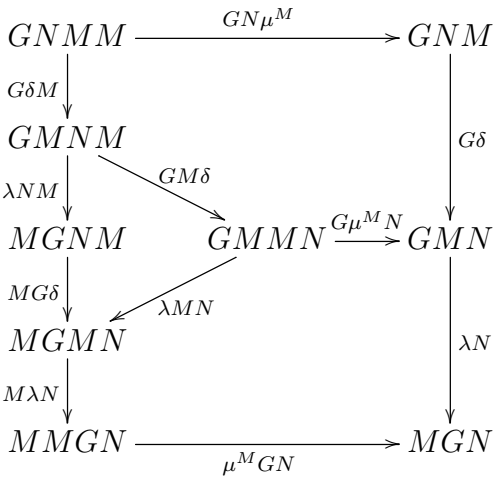


Lemma 8.17. *Given a distributive law $\lambda : GM \to MG$ of a functor G over the monad M and a distributive law $\delta : NM \to MN$ of monads, $\lambda N \cdot G\delta : GNM \to MGN$ is a δ-distributive law of the N-module $(GN, G\mu^N)$ over the monad M.*

Proof. From Example 2.42 we know that $(GN, G\mu^N)$ is an N-module. We check the three diagrams for a δ-distributive law. The one for η^M follows from the respective unit laws for the distributive laws δ and λ as shown in the following commutative diagram:



Likewise, the diagram for μ^M follows from the respective multiplication laws for the distributive laws δ and λ (and naturality of λ) as shown in the following commutative diagram:



For the last diagram for $G\mu^N$ consider the diagram

$$\begin{array}{ccc}
GNNM & \xrightarrow{G\mu^N M} & GNM \\
{\scriptstyle GN\delta}\downarrow & & \\
GNMN & & \downarrow{\scriptstyle G\delta} \\
{\scriptstyle G\delta N}\downarrow & & \\
GMNN & \xrightarrow{GM\mu^N} & GMN \\
{\scriptstyle \lambda NN}\downarrow & & \downarrow{\scriptstyle \lambda N} \\
MGNN & \xrightarrow[MG\mu^N]{} & MGN
\end{array}$$

which commutes by one of the multiplication laws for δ and by naturality of λ. □

Example 8.18. Using Lemma 8.17 we see that for the extensions $\lambda' : TM \to MT$ of the canonical distributive laws $\lambda : HM \to MH$ obtained in Proposition 8.12 ($M = \mathcal{P}^+$) and Corollary 8.7 ($M = (-)^E$), $\lambda T \cdot H\lambda' : HTM \to MHT$ is a λ'-distributive law.

Recall from Definition 2.43 the notion of an idealized monad.

Lemma 8.19. *Let $\delta : NM \to MN$ be a distributive law of the idealized monad $(N, \eta^N, \mu^N, \bar{N}, \bar{\mu}^N, \vartheta)$ over the monad (M, η^M, μ^M), and let $\bar{\delta} : \bar{N}M \to M\bar{N}$ be a δ-distributive law such that $M\vartheta \cdot \bar{\delta} = \delta \cdot \vartheta M$. Then the composite monad induced by δ is an idealized monad*

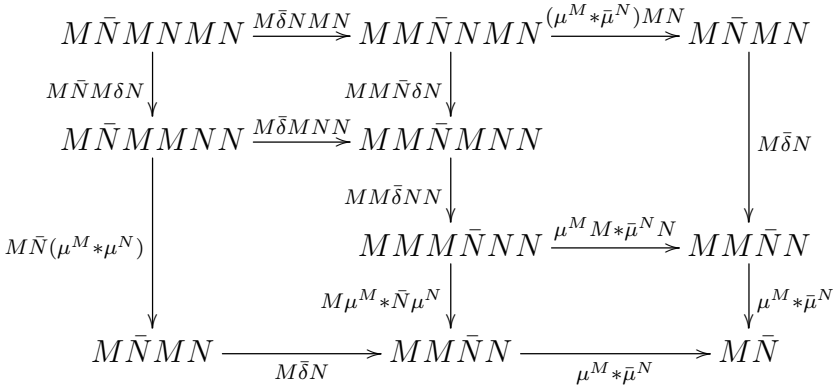
$$(MN, \eta^M * \eta^N, (\mu^M * \mu^N) \cdot M\delta N, M\bar{N}, (\mu^M * \bar{\mu}^N) \cdot M\bar{\delta}N, M\vartheta)\,.$$

Proof. We know from Lemma 8.15 that $(MN, \eta^M * \eta^N, (\mu^M * \mu^N) \cdot M\delta N)$ is a (composite) monad. We show that $(M\bar{N}, (\mu^M * \bar{\mu}^N) \cdot M\bar{\delta}N)$ is an MN-module: the first module diagram

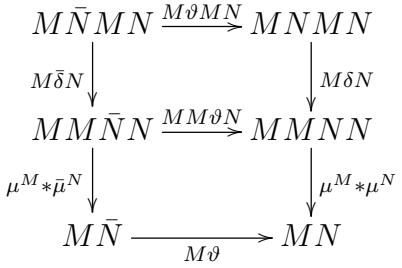
$$\begin{array}{ccccc}
M\bar{N} & \xrightarrow{M\bar{N}\eta^N} & M\bar{N}N & \xrightarrow{M\bar{N}\eta^M N} & M\bar{N}MN \\
 & \searrow & & \searrow_{M\eta^M\bar{N}N} & \downarrow{\scriptstyle M\bar{\delta}N} \\
 & & & & MM\bar{N}N \\
 & & & & \downarrow{\scriptstyle \mu^M * \bar{\mu}^N} \\
 & & & & M\bar{N}
\end{array}$$

commutes due to one of the laws for $\bar{\delta}$, one of the unit laws of the monad M and the first module diagram for the module $(\bar{N}, \bar{\mu}^N)$. The second module

diagram



commutes due to naturality of $\bar{\delta}$, the remaining two laws for $\bar{\delta}$, the multiplication law for M and the second module diagram for $(\bar{N}, \bar{\mu}^N)$. Finally we check that $M\vartheta$ is a module homomorphism: the diagram



commutes due to the assumed equation for $\bar{\delta}$ and δ, and due to $\vartheta : \bar{N} \to N$ being a module homomorphism. □

Example 8.20. Recall from Theorem 2.47 that the free CIM T on H is an idealized monad $(T, \eta^H, \mu^H, HT, H\mu^H, \tau)$. The distributive laws $\lambda' : TM \to MT$ from Example 8.18 arise from $\lambda : HM \to MH$ via the defining commutative diagram (8.2). Thus the preconditions of Lemma 8.19 are satisfied and $(MT, \eta^M * \eta^H, (\mu^M * \mu^H) \cdot M\lambda' T, MHT, (\mu^M * H\mu^H) \cdot M\lambda TT \cdot MH\lambda' T, M\tau)$ is an idealized composite monad.

Composite Monads $\mathcal{P}^+T$

Definition 8.21. A *weak completely iterative monad* (or *weak CIM*, for short) is an idealized monad where every guarded equation morphism has a solution (cf. Definition 2.45).

Definition 8.22. Let $e : X \to \mathcal{P}^+T(X + Y)$ be an equation morphism for the composite monad $\mathcal{P}^+T$ and let $f : \bar{X} \to X$ be a morphism. An equation

morphism $\bar{e} : \bar{X} \to T(\bar{X} + Y)$ for the monad T is said to be *over* e *via* f, if $(T(f + \mathrm{id}) \cdot \bar{e})(\bar{x}) \in (e \cdot f)(\bar{x})$ for all $\bar{x} \in \bar{X}$.

Theorem 8.23. *Let H be a finitary polynomial endofunctor on* Set*. For the extension $\lambda' : T\mathcal{P}^+ \to \mathcal{P}^+T$ of the canonical distributive law $\lambda : H\mathcal{P}^+ \to \mathcal{P}^+H$, the composite monad*

$$(\mathcal{P}^+T, \eta^+ * \eta^H, (\mu^+ * \mu^H) \cdot \mathcal{P}^+\lambda' T, \mathcal{P}^+HT, (\mu^+ * H\mu^H) \cdot \mathcal{P}^+\lambda TT \cdot \mathcal{P}^+H\lambda' T, \mathcal{P}^+\tau)$$

is a weak CIM.

Proof. We know from Example 8.20 that the six-tuple in the statement of the theorem is indeed an idealized monad.

We still have to check that guarded equation morphisms have a solution. To this end let $e : X \to \mathcal{P}^+T(X + Y)$ be any guarded equation morphism, i. e.

$$e = (X \xrightarrow{e'} (\mathcal{P}^+HT)(X + Y) + Y \xrightarrow{[\mathcal{P}^+\tau^H_{X+Y}, \eta^+_{T(X+Y)} \cdot \eta^H_{X+Y} \cdot \mathsf{inr}]} \mathcal{P}^+T(X + Y)) .$$

We prove that there is a solution $e^\dagger : X \to \mathcal{P}^+TY$ of e, i. e. the diagram

$$\begin{array}{ccc} X & \xrightarrow{\quad e^\dagger \quad} & \mathcal{P}^+TY \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle (\mu^+ * \mu^H)_Y} \\ & & \mathcal{P}^+\mathcal{P}^+TTY \\ & & \uparrow{\scriptstyle \mathcal{P}^+\lambda'_{TY}} \\ \mathcal{P}^+T(X + Y) & \xrightarrow[\mathcal{P}^+T[e^\dagger, \eta^+_{TY} \cdot \eta^H_Y]]{} & \mathcal{P}^+T\mathcal{P}^+TY \end{array} \tag{8.3}$$

commutes. To this end, we consider all equation morphisms $\bar{e}$ that are over e. Such equation morphisms have unique solutions $\bar{e}^\dagger : \bar{X} \to TY$, i. e. the diagram

$$\begin{array}{ccc} \bar{X} & \xrightarrow{\quad \bar{e}^\dagger \quad} & TY \\ {\scriptstyle \bar{e}}\downarrow & & \uparrow{\scriptstyle \mu^H_Y} \\ T(\bar{X} + Y) & \xrightarrow[T[\bar{e}^\dagger, \eta^H_Y]]{} & TTY \end{array}$$

commutes. In fact, since e is guarded, so are all the $\bar{e}$ as we see from $(T(f + \mathrm{id}) \cdot \bar{e})(\bar{x}) \in (e \cdot f)(\bar{x})$. Then the unique solutions are provided by the complete iterativity of T.

We define

$$e^\dagger(x) = \{\bar{e}^\dagger(\bar{x}) \mid \bar{e} \text{ is over } e \text{ via } f \text{ and } f(\bar{x}) = x\} \tag{8.4}$$

and prove that this is a solution of e. Observe that $e^\dagger$ is well-defined since the set $e^\dagger(x)$ is nonempty: there always exists a map $\bar{e}$ over e. To see this, choose $\bar{X} = X$, $f = \mathrm{id}$ and $\bar{e}(x) = t \in e(x)$. Observe in the definition of $\bar{e}$ that since every set $e(x) \in \mathcal{P}^+T(X+Y)$ is nonempty, there always exists a tree $t \in e(x)$. Then clearly $\bar{e}$ is over e via f.

We prove that $e^\dagger$ is a solution, i. e. diagram (8.3) commutes. We do so pointwise, i. e. we show that for every $x \in X$ we have $e^\dagger(x) = ((\mu^+ * \mu^H)_Y \cdot \mathcal{P}^+\lambda'_{TY} \cdot \mathcal{P}^+T[e^\dagger, \eta^+_{TY} \cdot \eta^H_Y] \cdot e)(x)$. We prove the two subset inclusions.

$\subseteq$ Let $t \in e^\dagger(x)$. By (8.4) this means that there exists $\bar{e} : \bar{X} \to T(\bar{X}+Y)$ over e via some $f : \bar{X} \to X$ and $\bar{x} \in \bar{X}$ such that $f(\bar{x}) = x$ and $t = \bar{e}^\dagger(\bar{x})$. By the solution diagram for $\bar{e}$ we can rewrite the latter equation to $t = (\mu^H_Y \cdot T[\bar{e}^\dagger, \eta^H_Y] \cdot \bar{e})(\bar{x})$. We argue that from this it follows $t \in (\mathcal{P}^+\mu^H_Y \cdot \lambda'_{TY} \cdot T[e^\dagger \cdot f, \eta^+_{TY} \cdot \eta^H_Y] \cdot \bar{e})(\bar{x})$: in both formulas, $\bar{e}(\bar{x})$ gives the same tree that may have several nodes labeled by elements from $\bar{X}$ or Y. Whereas the nodes with labels in Y are then considered as subtrees in the original equation, they are considered as singleton sets of a subtree in the new formula. And whereas the nodes with labels in $\bar{X}$ are replaced by trees according to $\bar{e}^\dagger$ in the original equation, they are replaced by sets of trees according to $e^\dagger \cdot f$ in the new formula. Notice that these sets contain the corresponding trees that are substituted in the original equation since for every $\bar{z} \in \bar{X}$ with $z = f(\bar{z})$ we have

$$\bar{e}^\dagger(\bar{z}) \in e^\dagger(z) = (e^\dagger \cdot f)(\bar{z})$$

by (8.4) since $\bar{e}$ is over e via f. Now in the new formula λ' makes the tree with sets of trees in its leaves the set of all trees where one tree per leaf is chosen from the (nonempty) set (see Lemma 8.10), so clearly the tree with trees in its leaves from the original equation appears in this set. Finally, μ^H glues this tree with trees in its leaves into one tree in both cases, so the desired new formula holds. Now we easily rewrite it to finish this direction of the proof:

$$\begin{aligned}
& t \in (\mathcal{P}^+\mu^H_Y \cdot \lambda'_{TY} \cdot T[e^\dagger \cdot f, \eta^+_{TY} \cdot \eta^H_Y] \cdot \bar{e})(\bar{x}) \\
\iff\ & t \in (\mathcal{P}^+\mu^H_Y \cdot \lambda'_{TY} \cdot T[e^\dagger, \eta^+_{TY} \cdot \eta^H_Y] \cdot T(f + \mathrm{id}) \cdot \bar{e})(\bar{x}) \\
\implies\ & t \in (\mu^+_{TY} \cdot \mathcal{P}^+\mathcal{P}^+\mu^H_Y \cdot \mathcal{P}^+\lambda'_{TY} \cdot \mathcal{P}^+T[e^\dagger, \eta^+_{TY} \cdot \eta^H_Y] \cdot e \cdot f)(\bar{x}) \\
\iff\ & t \in ((\mu^+ * \mu^H)_Y \cdot \mathcal{P}^+\lambda'_{TY} \cdot \mathcal{P}^+T[e^\dagger, \eta^+_{TY} \cdot \eta^H_Y] \cdot e)(x)
\end{aligned}$$

$\supseteq$ Let $t \in ((\mu^+ * \mu^H)_Y \cdot \mathcal{P}^+\lambda'_{TY} \cdot \mathcal{P}^+T[e^\dagger, \eta^+_{TY} \cdot \eta^H_Y] \cdot e)(x)$. Since t is an element of a union (performed by μ^+), it must be an element of one of the given sets, i. e. there exists $s \in e(x)$ such that $t \in (\mathcal{P}^+\mu^H_Y \cdot \lambda'_{TY} \cdot T[e^\dagger, \eta^+_{TY} \cdot \eta^H_Y])(s)$. Thus t is constructed from s as follows: every x_i-labeled leaf is replaced by the set $e^\dagger(x_i) = \{\bar{e}^\dagger(\bar{x}_i) \mid \bar{e} \text{ over } e \text{ via } f \text{ and } f(\bar{x}_i) = x_i\}$ from which a tree is

then chosen to be the subtree t_i of t replacing the x_i-labeled leaf (recall the meaning of λ' and μ^H). This means that we have for every x_i-labeled leaf some $\bar{e}_i$ over e via f_i with $f_i(\bar{x}_i) = x_i$ and $\bar{e}_i^\dagger(\bar{x}_i) = t_i$. We form the disjoint union of all f_i and add $f(\bar{x}) = x$ to obtain f. Also, we form the disjoint union of all $\bar{e}_i$ and add $\bar{e}(\bar{x}) = s'$ where $T(f + \mathrm{id})(s') = s$ to obtain $\bar{e}$. The existence of such an s' is clear since we can choose s with the x_i replaced by the $\bar{x}_i$. Then it is easy to see that $\bar{e}$ is over e via f and that $\bar{e}^\dagger(\bar{x}_i) = \bar{e}_i^\dagger(\bar{x}_i)$. We finally see that

$$\begin{aligned} t &= s[t_i/x_i] = s'[t_i/\bar{x}_i] = s'[\bar{e}_i^\dagger(\bar{x}_i)/\bar{x}_i] = s'[\bar{e}^\dagger(\bar{x}_i)/\bar{x}_i] \\ &= (\mu_Y^H \cdot T[\bar{e}^\dagger, \eta_Y^H])(s') \\ &= (\mu_Y^H \cdot T[\bar{e}^\dagger, \eta_Y^H] \cdot \bar{e})(\bar{x}) \\ &= \bar{e}^\dagger(\bar{x})\,. \end{aligned}$$

Since $\bar{e}$ is over e via f and $f(\bar{x}) = x$, we obtain $t \in e^\dagger(x)$ as desired. □

Remark 8.24. The part of the proof of Theorem 8.23 showing that all guarded equation morphisms have a solution even works for the (non-guarded) equation morphisms $e : X \to \mathcal{P}^+T(X + Y)$ that factor as follows:

$$e \equiv (X \xrightarrow{e'} \mathcal{P}^+(HX + Y) \xrightarrow{\mathcal{P}^+(\kappa_X^H + \eta_Y^H)} \mathcal{P}^+(TX + TY) \xrightarrow{\mathcal{P}^+\mathsf{can}} \mathcal{P}^+T(X + Y))\,.$$

The reason is that although the equation morphism e is not necessarily guarded, the equation morphisms $\bar{e} : X \to T(X + Y)$ that are over e via some f always are; the rest of the proof remains the same.

Definition 8.25. We shall call morphisms e which factor as shown in Remark 8.24 *almost guarded*.

Observe (similar as above Proposition 7.33) that for every set Y, $\mathcal{P}^+Y$ carries the partial order $\subseteq$. This extends pointwise to a partial order $\leq$ on all sets $\mathsf{Set}(X, \mathcal{P}^+Y)$ of functions. In this sense we use the term "greatest solution/homomorphism" in the following lemma and in Lemma 8.34 below as well as in Section 8.5. We also make use of the term "cutting of a tree". We think that our use of it should be unambiguous; however, for one possible precise definition of a cutting of a tree at some level we refer the reader to [AM06].

Lemma 8.26. *The solutions $e^\dagger$ of (almost) guarded equation morphisms e from the proof of Theorem 8.23 and Remark 8.24 are greatest solutions; moreover, for all solutions s of an (almost) guarded equation morphism e the sets of all finite cuttings of the trees from $e^\dagger(x)$ and $s(x)$ are the same for every $x \in X$.*

Proof. Let $e : X \to \mathcal{P}^+T(X+Y)$ be an (almost) guarded equation morphism. Recall from the proof of Theorem 8.23 that it has the solution $e^\dagger : X \to \mathcal{P}^+TY$ given by $e^\dagger(x) = \{\bar{e}^\dagger(\bar{x}) \mid \bar{e} \text{ is over } e \text{ via } f \text{ and } f(\bar{x}) = x\}$.

(1) $e^\dagger$ is a greatest solution. Assume s to be any solution of e, i. e. diagram (8.3) commutes with $e^\dagger$ replaced by s. Then we have to show $s \leq e^\dagger$, i. e. $s(x) \subseteq e^\dagger(x)$ for every $x \in X$.
Let $t \in s(x)$; according to diagram (8.3) this means $t \in ((\mu^+ * \mu^H)_Y \cdot \mathcal{P}^+\lambda'_{TY} \cdot \mathcal{P}^+T[s, \eta^+_{TY} \cdot \eta^H_Y] \cdot e)(x)$. If t is a singleton tree labeled by some $y \in Y$ then $e(x)$ must contain this singleton tree since e is guarded or almost guarded. If t is a singleton tree labeled by a constant operation, the same is true. If the root of t is labeled by some operation symbol with arity ≥ 1, then we have a tree in $e(x)$ with at most countably many variables x_i (or parameters from Y) in its leaves (to have no leaves labeled in X is also possible), which has depth ≥ 1 and which equals t except that t has subtrees t_i instead of the x_i where $t_i \in s(x_i)$. Since the x_i have depth at least 1 in this tree, and we have the original condition again, we can apply our argument inductively.
Let us now construct an equation morphism as follows: $\bar{X}$ initially contains a variable $\bar{x}$ and we let $f(\bar{x}) = x$. In every step of the above induction we set for all variables $\bar{z} \in \bar{X}$ for which $\bar{e}$ is still undefined $\bar{e}(\bar{z})$ to be the corresponding tree from $e(z)$, $z = f(\bar{z}) \in X$ we found, where every variable z_i is replaced by a fresh one $\bar{z}_i$ we add to $\bar{X}$; according to this replacement we define f on the newly introduced variables by $f(\bar{z}_i) = z_i$. At step n it is already clear that the solution $\bar{e}^\dagger(\bar{x})$ of $\bar{e}$ (which is not completely defined yet) must equal t at least up to level n. Thus the solution $\bar{e}^\dagger(\bar{x})$ of the completely defined $\bar{e}$ equals t up to every level which means $\bar{e}^\dagger(\bar{x}) = t$.
Furthermore, we see that $\bar{e}$ is over e via f since by its definition we have $(T(f+Y) \cdot \bar{e})(\bar{x}) \in (e \cdot f)(\bar{x})$ for every $\bar{x} \in \bar{X}$. Thus we finally have $t \in e^\dagger(x)$.

(2) For all solutions of an (almost) guarded equation morphism the set of all finite cuttings of the trees from the solution is the same. Let $t \in e^\dagger(x) = ((\mu^+ * \mu^H)_Y \cdot \mathcal{P}^+\lambda'_{TY} \cdot \mathcal{P}^+T[e^\dagger, \eta^+_{TY} \cdot \eta^H_Y] \cdot e)(x)$ be a tree from the greatest solution. The (almost) guardedness of e and the concrete actions of λ' (see Lemma 8.10), μ^+ (see Example 6.1(4)) and μ^H (flattening of trees with trees in their leaves) imply that there exists a tree $t_0 \in e(x)$ all of whose leaves labeled by variables $x_i \in X$ are at levels ≥ 1, and t arises from t_0 by substituting each x_i by some tree from $e^\dagger(x_i)$. Notice that this includes the special case where t_0 has zero leaves labeled by variables from X.
Now assume s to be any solution of e, i. e. $s(x) = ((\mu^+ * \mu^H)_Y \cdot \mathcal{P}^+\lambda'_{TY} \cdot \mathcal{P}^+T[s, \eta^+_{TY} \cdot \eta^H_Y] \cdot e)(x)$. By the concrete actions of λ', μ^+ and μ^H and since $t_0 \in e(x)$, this means that $s(x)$ contains all trees that arise by substituting in t_0 the x_i with a tree from $s(x_i)$. And there is at least one such tree in $s(x)$ since the $s(x_i)$ are all nonempty. This tree must have the same shape as t up

to the substitutions, i. e. at least up to level 0, because λ' does not change this shape. Thus we have a tree in $s(x)$ whose cutting at level 0 equals the cutting of t at level 0.
We can repeat our argument at the x_i instead of x and so on to see that for every $n \in \mathbb{N}$ we have a tree in $s(x)$ whose cutting at level n equals the cutting of t at level n. Thus we have proved that every finite cutting of a tree from the greatest solution can also be obtained as a cutting of a tree from any solution. This concludes our proof since the other subset inclusion is immediate: $e^\dagger$ is the greatest solution the finite cuttings of whose trees are a superset of the finite cuttings of any other solution. □

Composite Monads T^E

We shall not need it in the sequel, but for the sake of completeness we prove

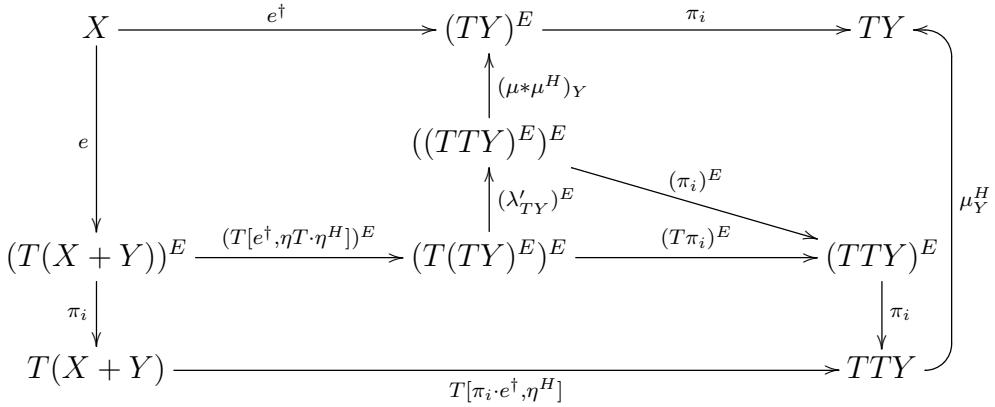
Theorem 8.27. *Let H be any iteratable endofunctor and let $((-)^E, \eta, \mu)$ be the environment monad. For the extension $\lambda' : T(-)^E \to T^E$ of the canonical distributive law $\lambda : H(-)^E \to H^E$, the composite monad*

$$(T^E, \eta * \eta^H, (\mu * \mu^H) \cdot (\lambda' T)^E, (HT)^E, (\mu * H\mu^H) \cdot (\lambda TT)^E \cdot (H\lambda' T)^E, \tau^E)$$

is a CIM.

Proof. The six-tuple in the statement is indeed an idealized monad, see Example 8.20.

Let $e : X \to (T(X+Y))^E$ be an equation morphism. We prove that $e^\dagger : X \to (TY)^E$ is a solution of e precisely if $\pi_i \cdot e^\dagger : X \to TY$ is a solution of $\pi_i \cdot e : X \to T(X+Y)$ for all $i \in E$. To this end, consider the following diagram:



The upper right-hand part commutes since $\pi_i \cdot \mu = \pi_i \cdot (\pi_i)^E$ (see above Lemma 7.24), the triangle commutes by the concrete action of λ' (forming

the i-th tree by using the i-th projection in each leaf) and the lower square commutes since $\pi_i \cdot \eta = \mathrm{id}$ (see above Lemma 7.24) and by naturality of π_i. Thus the remaining upper left-hand square commutes precisely if the outside commutes for every $i \in E$ as desired (for one direction one uses that the π_i can be jointly canceled).

Now every guarded equation morphism

$$e = (X \xrightarrow{e'} (HT(X+Y))^E + Y \xrightarrow{[\tau^E_{X+Y}, (\eta * \eta^H)_{X+Y} \cdot \mathsf{inr}]} (T(X+Y))^E)$$

has a unique solution: in fact, for every $i \in E$ the equation morphisms $\pi_i \cdot e = \pi_i \cdot [\tau^E_{X+Y}, (\eta * \eta^H)_{X+Y} \cdot \mathsf{inr}] \cdot e' = [\tau_{X+Y} \cdot \pi_i, \eta^H_{X+Y} \cdot \mathsf{inr}] \cdot e'$ are guarded by $(\pi_i + \mathrm{id}) \cdot e' : X \to HT(X+Y) + Y$ and thus have a unique solution since T is the free CIM. From the equivalence proved above we conclude that e has a unique solution. □

Observation 8.28. It is trivial to observe that the idealized composite monads from Example 8.20 for the special case $E = 1$ (i. e. $M = \mathrm{Id}$) are CIMs: since the canonical distributive laws and their extensions are the identities $\lambda = \mathrm{id}$ and $\lambda' = \mathrm{id}$, these monads collapse to the free CIMs T on H.

8.3 (Weakly) Final Coalgebras

As in Section 8.2 we start with a part which is common for all effects or monads under consideration. In this first part we explain for which endofunctor on which category we then investigate the final coalgebra (or weakly final coalgebra in case of "nonempty nondeterminism"). The (weakly) final coalgebras play an important role in defining certain morphisms and proving them to be monad morphisms (in Section 8.4) and in proving guarded RPS's with effects to have unique uninterpreted solutions (or canonical solutions in case of "nonempty nondeterminism", in Section 8.5).

This first part even works for arbitrary categories $\mathcal{C}$ with finite coproducts. Let us denote by $[\mathcal{C}, \mathcal{C}]$ the category of all endofunctors on $\mathcal{C}$ (the morphisms are the natural transformations between them). This category also has finite coproducts which are formed componentwise. Any functor $H : \mathcal{C} \to \mathcal{C}$ gives rise to the functor $\mathcal{H} : [\mathcal{C}, \mathcal{C}] \to [\mathcal{C}, \mathcal{C}]$ defined on objects (i. e. functors G) and morphisms (i. e. natural transformations $\alpha : G \to G'$) by

$$\mathcal{H}G = HG + \mathrm{Id} \quad \text{and} \quad \mathcal{H}\alpha = H\alpha + \mathrm{id}\,.$$

And any monad (M, η^M, μ^M) on $\mathcal{C}$ gives rise to a monad $(\mathcal{M}, \eta^{\mathcal{M}}, \mu^{\mathcal{M}})$ on $[\mathcal{C}, \mathcal{C}]$ as follows: the functor $\mathcal{M}$ is defined by

$$\mathcal{M}G = MG \quad \text{and} \quad \mathcal{M}\alpha = M\alpha\,,$$

and the G-components of unit and multiplication are given by $\eta^{\mathcal{M}}_G = \eta^M G$ and $\mu^{\mathcal{M}}_G = \mu^M G$. The monad laws follow straight from the ones for (M, η^M, μ^M).

Lemma 8.29. *Any distributive law λ of a functor H over a monad M in $\mathcal{C}$ induces a distributive law Λ of the functor $\mathcal{H}$ over the monad $\mathcal{M}$ in $[\mathcal{C},\mathcal{C}]$.*

Proof. For every object from $[\mathcal{C},\mathcal{C}]$ (i. e. every functor $G : \mathcal{C} \to \mathcal{C}$) we define $\Lambda_G = \mathsf{can} \cdot (\lambda G + \eta^M)$. Naturality of Λ is proved by the commutative diagram

$$\begin{array}{ccccccc}
 & & & \overset{\Lambda_G}{\frown} & & & \\
\mathcal{HM}G = & HMG + \mathrm{Id} & \xrightarrow{\lambda G + \eta^M} & MHG + M & \xrightarrow{\mathsf{can}} & M(HG + \mathrm{Id}) & = \mathcal{MH}G \\
 & \downarrow{\scriptstyle \mathcal{HM}\alpha = HM\alpha + \mathrm{id}} & & \downarrow{\scriptstyle MH\alpha + M\mathrm{id}} & & \downarrow{\scriptstyle M(H\alpha + \mathrm{id}) = \mathcal{MH}\alpha} & \\
\mathcal{HM}G' = & HMG' + \mathrm{Id} & \xrightarrow[\lambda G' + \eta^M]{} & MHG' + M & \xrightarrow[\mathsf{can}]{} & M(HG' + \mathrm{Id}) & = \mathcal{MH}G' \\
 & & & \underset{\Lambda'_G}{\smile} & & &
\end{array}$$

for every morphism from $[\mathcal{C},\mathcal{C}]$ (i. e. every natural transformation $\alpha : G \to G'$ in $\mathcal{C}$): the left-hand part commutes by naturality of λ, and the right-hand part by naturality of can in $\mathcal{C}$. The two laws for Λ are easily checked componentwise for every object of $[\mathcal{C},\mathcal{C}]$ (i. e. for every functor G) in

$$\begin{aligned}
\Lambda_G \cdot \mathcal{H}\eta^{\mathcal{M}}_G &= \mathsf{can} \cdot (\lambda G + \eta^M) \cdot (H\eta^M G + \mathrm{id}) \\
&= \mathsf{can} \cdot (\eta^M HG + \eta^M) \\
&= \eta^M (HG + \mathrm{Id}) \\
&= \eta^{\mathcal{M}}_{\mathcal{H}G}
\end{aligned}$$

and

$$\begin{aligned}
\Lambda_G \cdot \mathcal{H}\mu^{\mathcal{M}}_G &= \mathsf{can} \cdot (\lambda G + \eta^M) \cdot (H\mu^M G + \mathrm{id}) \\
&= \mathsf{can} \cdot (\mu^M HG + \mu^M) \cdot (M\lambda G + M\eta^M) \cdot (\lambda MG + \eta^M) \\
&= \mu^M (HG + \mathrm{Id}) \cdot M\mathsf{can} \cdot \mathsf{can} \cdot (M\lambda G + M\eta^M) \cdot (\lambda MG + \eta^M) \\
&= \mu^M (HG + \mathrm{Id}) \cdot M\mathsf{can} \cdot M(\lambda G + \eta^M) \cdot \mathsf{can} \cdot (\lambda MG + \eta^M) \\
&= \mu^{\mathcal{M}}_{\mathcal{H}G} \cdot \mathcal{M}\Lambda_G \cdot \Lambda_{\mathcal{M}G} \,.
\end{aligned}$$

□

It follows from Proposition 6.6 that $\mathcal{H}$ lifts to a functor $\bar{\mathcal{H}}$ on $[\mathcal{C},\mathcal{C}]_{\mathcal{M}}$. By Remark 6.7 $\bar{\mathcal{H}}$ is given on objects (i. e. functors G) and morphisms (i. e. natural transformations $\alpha : G \to MG'$) by

$$\bar{\mathcal{H}}G = HG + \mathrm{Id} \quad \text{and} \quad \bar{\mathcal{H}}\alpha = \mathsf{can} \cdot (\lambda G' + \eta^M) \cdot (H\alpha + \mathrm{id}) \,.$$

Observe that writing the X-component of the latter equation in the Kleisli category $\mathcal{C}_M$ yields $(\bar{\mathcal{H}}\alpha)_X = \bar{H}\alpha_X + \mathrm{id}_X$, where $\bar{H}$ is the lifting of H to $\mathcal{C}_M$ induced by λ (see Proposition 6.6).

We now turn back to the category $\mathcal{C} = \mathsf{Set}$ and the monads M from Example 6.1.

Final Coalgebras for $\bar{\mathcal{H}}$ under Assumption 7.16

Let H be finitary and recall the natural transformations $\phi : HF \to F$ and $\eta : \mathrm{Id} \to F$ given componentwise by the free H-algebras and their universal morphisms. Notice that $[\phi, \eta] : \mathcal{H}F = HF + \mathrm{Id} \to F$ is the initial $\mathcal{H}$-algebra; the proof is simple and therefore left to the reader. Recall that $J : \mathsf{Set} \to \mathsf{Set}_M$ denotes the inclusion functor from the Kleisli adjunction.

Theorem 8.30 (cf. [HJS07])**.** *Under Assumption 7.16, $F \xrightarrow{J[\phi,\eta]^{-1}} \bar{\mathcal{H}}F$ is a final $\bar{\mathcal{H}}$-coalgebra.*

Proof. We first prove that from Assumption 7.16 on H and M it follows that the same assumption holds true for $\mathcal{H}$ and $\mathcal{M}$:

1. Since H is finitary, so is $\mathcal{H}$. This follows from the fact that colimits in the functor category $[\mathsf{Set}, \mathsf{Set}]$ are taken objectwise.

2. We have seen in Lemma 8.29 that λ induces a distributive law Λ of the functor $\mathcal{H}$ over the monad $\mathcal{M}$.

3. A CPO $\sqsubseteq$ on every hom-set $[\mathsf{Set}, \mathsf{Set}]_{\mathcal{M}}(G, G')$ is given componentwise by the CPOs $\leq$ on $\mathsf{Set}_M(GX, G'X)$: the morphisms $\alpha : G \nrightarrow G'$ from $[\mathsf{Set}, \mathsf{Set}]_{\mathcal{M}}$ are natural transformations $\alpha : G \to MG'$ in Set whose components $\alpha_X : GX \to MG'X$ are morphisms $\alpha_X : GX \nrightarrow G'X$ in Set_M. We define $\alpha \sqsubseteq \beta$ iff $\alpha_X \leq \beta_X$ for all objects X of Set, then the least element and the suprema of ω-chains are simply formed componentwise. That $[\mathsf{Set}, \mathsf{Set}]_{\mathcal{M}}$ is CPO-enriched and composition is left-strict follows along this definition from the fact that Set_M is CPO-enriched and composition is left-strict.

4. Local monotonicity of $\bar{\mathcal{H}}$ w. r. t. $\sqsubseteq$ follows from local monotonicity of $\bar{H}$ w. r. t. $\leq$. To see this, consider consider $\alpha \sqsubseteq \beta$ with $\alpha, \beta \in [\mathsf{Set}, \mathsf{Set}]_{\mathcal{M}}(G, G')$, i. e. $\alpha_X \leq \beta_X$ for all X. From local monotonicity of $\bar{H}$ we have $\bar{H}\alpha_X \leq \bar{H}\beta_X$ for all X and due to monotonicity of the coproduct in Set_M we conclude $\bar{H}\alpha_X + \mathrm{id}_X \leq \bar{H}\beta_X + \mathrm{id}_X$ (written in Set_M). It follows (still written in Set_M)

$$(\bar{\mathcal{H}}\alpha)_X = \bar{H}\alpha_X + \mathrm{id}_X \leq \bar{H}\beta_X + \mathrm{id}_X = (\bar{\mathcal{H}}\beta)_X$$

for all sets X which means $\bar{\mathcal{H}}\alpha \sqsubseteq \bar{\mathcal{H}}\beta$.

Since $[\phi, \eta]$ is the initial $\mathcal{H}$-algebra, the proof of Theorem 7.19 (which can be found in [HJS07]) can be carried out in the functor category $[\mathsf{Set}, \mathsf{Set}]$ instead of Set, replacing H by $\mathcal{H}$ and M by $\mathcal{M}$. Thus we obtain the desired result. □

Corollary 8.31. *Let λ be the canonical distributive law of an analytic functor H over one of the monads $\mathrm{Id}+1$, $\mathcal{P}$ or $\mathcal{D}$ from Theorem 6.19. Then $J[\phi, \eta]^{-1}$ is the final $\bar{\mathcal{H}}$-coalgebra.*

Proof. Recall from Example 7.18 that for the canonical distributive law of an analytic functor H over one of the monads $M = \mathrm{Id}+1$, $M = \mathcal{P}$ or $M = \mathcal{D}$ Assumption 7.16 is satisfied. Then the statement follows from Theorem 8.30. □

Weakly Final Coalgebras for $\bar{\mathcal{H}}$ in case $M = \mathcal{P}^+$

Definition 8.32. Given a functor G, a G-coalgebra (W, w) is called *weakly final* if for every G-coalgebra (S, s) there exists a homomorphism from (S, s) to (W, w).

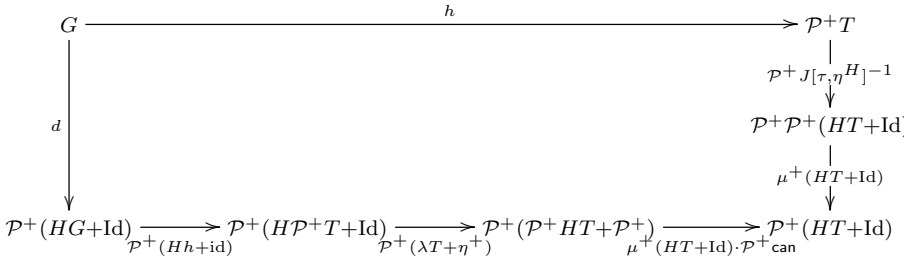
Let H be finitary polynomial; then H is iteratable. Recall from [MM06], Theorem 5.1 that $[\tau, \eta^H]^{-1} : T \to HT + \mathrm{Id} = \mathcal{H}T$ is the final coalgebra for $\mathcal{H}$, where τ_X is the free CIA on X with universal arrow η^H_X.

Let furthermore $\bar{\mathcal{H}}$ arise from the canonical distributive law λ of H over $\mathcal{P}^+$. As we shall prove next, $J[\tau, \eta^H]^{-1}$ is a weakly final $\bar{\mathcal{H}}$-coalgebra. This implies that unlike under Assumption 7.16 (see Theorem 8.30), $J[\phi, \eta^H]^{-1}$ is not the final $\bar{\mathcal{H}}$-coalgebra since it is not difficult to see that it is no quotient of the weakly final one $J[\tau, \eta^H]^{-1}$. Also $J[\tau, \eta^H]^{-1}$ is not the final $\bar{\mathcal{H}}$-coalgebra: if it was, this result would propagate through the rest of this chapter and would yield unique instead of greatest solutions in Theorem 8.60. But as demonstrated on the NNRPS (II.1) in the introduction to Part II, there may be more than one solution in general.

The proof of weak finality of $J[\tau, \eta^H]^{-1}$ exploits that $\mathcal{P}^+T$ is a weak CIM (see Theorem 8.23) and that even almost guarded equation morphisms can be solved (see Remark 8.24). To this end, it establishes a relation between solutions of equation morphisms for the weak CIM $\mathcal{P}^+T$ and homomorphisms into the $\bar{\mathcal{H}}$-coalgebra $J[\tau, \eta^H]^{-1}$.

Theorem 8.33. *Let λ be the canonical distributive law of a finitary polynomial functor H over the nonempty powerset monad $M = \mathcal{P}^+$ from Example 6.14(4). Then $T \xrightarrow{J[\tau,\eta^H]^{-1}\bullet} \bar{\mathcal{H}}T$ is a weakly final $\bar{\mathcal{H}}$-coalgebra.*

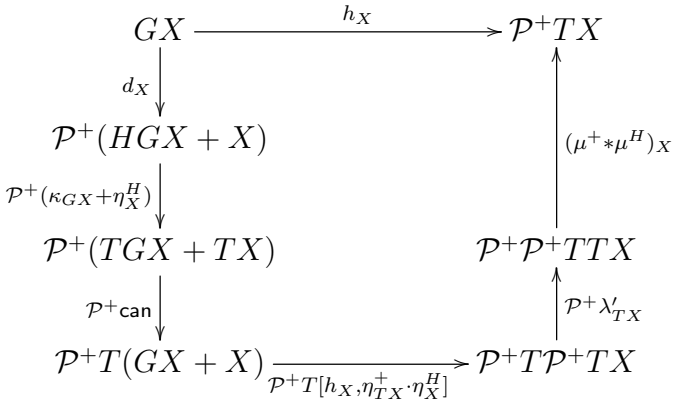
Proof. Given any $\bar{\mathcal{H}}$-coalgebra $d : G \to \mathcal{P}^+(HG + \mathrm{Id})$ we prove that there is a homomorphism $h : G \to \mathcal{P}^+T$ into the coalgebra $J[\tau, \eta^H]^{-1} : T \to \mathcal{P}^+(HT + \mathrm{Id})$, i. e. we prove that the diagram



(8.5)

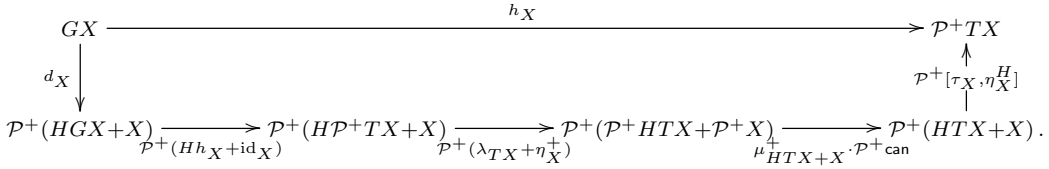
in $[\mathsf{Set}, \mathsf{Set}]$ commutes. It suffices to prove that (1) for every set X there is a map $h_X : GX \to \mathcal{P}^+TX$ such that the X-component of the diagram commutes, and (2) the h_X form a natural transformation.

(1) We consider the X-component of diagram (8.5) and show that it commutes precisely if the diagram



(8.6)

commutes. It follows from Theorem 8.23 and Remark 8.24 that there is an h_X making diagram (8.6) commute.
So we are left to show the equivalence of the two diagrams (8.5) and (8.6) above. We start by observing that the right-hand outside of diagram (8.5) is $\mathcal{P}^+[\tau, \eta]^{-1}$ and that we can reverse this isomorphism. So the X-component of diagram (8.5) equivalently is



We rewrite the last two arrows of the lower part as in

$$
\begin{aligned}
&\mathcal{P}^+[\tau_X, \eta^H_X] \cdot \mu^+_{HTX+X} \cdot \mathcal{P}^+\mathsf{can} \\
&= \mu^+_{TX} \cdot \mathcal{P}^+\mathcal{P}^+[\mu^H_X \cdot \kappa_{TX}, \mu^H_X \cdot \eta^H_{TX} \cdot \eta^H_X] \cdot \mathcal{P}^+\mathsf{can} \\
&\qquad \text{(by naturality of } \mu^+\text{, Lemma 2.49 and one of the unit laws for } T) \\
&= \mu^+_{TX} \cdot \mathcal{P}^+\mathcal{P}^+\mu^H_X \cdot \mathcal{P}^+\mathcal{P}^+[\kappa_{TX}, \eta^H_{TX} \cdot \eta^H_X] \cdot \mathcal{P}^+\mathsf{can} \\
&= (\mu^+ * \mu^H)_X \cdot \mathcal{P}^+\mathcal{P}^+[\mathrm{id}_{TTX}, \eta^H_{TX} \cdot \eta^H_X] \cdot \mathcal{P}^+\mathsf{can} \cdot \mathcal{P}^+(\mathcal{P}^+\kappa_{TX} + \mathrm{id}_{\mathcal{P}^+X}) \\
&= (\mu^+ * \mu^H)_X \cdot \mathcal{P}^+[\mathrm{id}_{\mathcal{P}^+TTX}, \mathcal{P}^+\eta^H_{TX} \cdot \mathcal{P}^+\eta^H_X] \cdot \mathcal{P}^+(\mathcal{P}^+\kappa_{TX} + \mathrm{id}_{\mathcal{P}^+X}) \\
&= (\mu^+ * \mu^H)_X \cdot \mathcal{P}^+[\mathrm{id}_{\mathcal{P}^+TTX}, \lambda'_{TX} \cdot \eta^H_{\mathcal{P}^+TX} \cdot \mathcal{P}^+\eta^H_X] \cdot \mathcal{P}^+(\mathcal{P}^+\kappa_{TX} + \mathrm{id}_{\mathcal{P}^+X}) \\
&\qquad \text{(by one of the laws for } \lambda')
\end{aligned}
$$

to obtain the diagram

$$
\begin{array}{ccccccc}
GX & & & \xrightarrow{h_X} & & & \mathcal{P}^+TX \\
{\scriptstyle d_X}\downarrow & & & & & & \uparrow{\scriptstyle (\mu^+ * \mu^H)_X} \\
\mathcal{P}^+(HGX+X) & \xrightarrow[\mathcal{P}^+(Hh_X+\mathrm{id}_X)]{} & \mathcal{P}^+(H\mathcal{P}^+TX+X) & \xrightarrow[\mathcal{P}^+(\mathcal{P}^+\kappa_{TX}\cdot\lambda_{TX}+\eta^+_X)]{} & \mathcal{P}^+(\mathcal{P}^+TTX+\mathcal{P}^+X) & \xrightarrow[\mathcal{P}^+[\mathrm{id}_{\mathcal{P}^+TTX},\lambda'_{TX}\cdot\eta^H_{\mathcal{P}^+TX}\cdot\mathcal{P}^+\eta^H_X]]{} & \mathcal{P}^+\mathcal{P}^+TTX
\end{array}
$$

We rewrite the second, third and fourth arrow of the the lower part as in

$$
\begin{aligned}
&\mathcal{P}^+[\mathrm{id}_{\mathcal{P}^+TTX}, \lambda'_{TX} \cdot \eta^H_{\mathcal{P}^+TX} \cdot \mathcal{P}^+\eta^H_X] \cdot \mathcal{P}^+(\mathcal{P}^+\kappa_{TX} \cdot \lambda_{TX} + \eta^+_X) \\
&\cdot \mathcal{P}^+(Hh_X + \mathrm{id}_X) \\
&= \mathcal{P}^+[\mathrm{id}_{\mathcal{P}^+TTX}, \lambda'_{TX} \cdot \eta^H_{\mathcal{P}^+TX} \cdot \mathcal{P}^+\eta^H_X] \cdot \mathcal{P}^+(\lambda'_{TX} \cdot \kappa_{\mathcal{P}^+TX} + \eta^+_X) \\
&\cdot \mathcal{P}^+(Hh_X + \mathrm{id}_X) \qquad \text{(by Remark 8.13)} \\
&= \mathcal{P}^+[\mathrm{id}_{\mathcal{P}^+TTX}, \lambda'_{TX} \cdot \eta^H_{\mathcal{P}^+TX} \cdot \mathcal{P}^+\eta^H_X] \cdot \mathcal{P}^+(\lambda'_{TX} + \eta^+_X) \cdot \mathcal{P}^+(Th_X + \mathrm{id}_X) \\
&\cdot \mathcal{P}^+(\kappa_{GX} + \mathrm{id}_X) \qquad \text{(by naturality of } \kappa) \\
&= \mathcal{P}^+\lambda'_{TX} \cdot \mathcal{P}^+[\mathrm{id}_{T\mathcal{P}^+TX} + \eta^H_{\mathcal{P}^+TX} \cdot \mathcal{P}^+\eta^H_X \cdot \eta^+_X] \cdot \mathcal{P}^+(Th_X + \mathrm{id}_X) \\
&\cdot \mathcal{P}^+(\kappa_{GX} + \mathrm{id}_X) \\
&= \mathcal{P}^+\lambda'_{TX} \cdot \mathcal{P}^+[\mathrm{id}_{T\mathcal{P}^+TX} + T\mathcal{P}^+\eta^H_X \cdot T\eta^+_X] \cdot \mathcal{P}^+(Th_X + \mathrm{id}_{TX}) \\
&\cdot \mathcal{P}^+(\kappa_{GX} + \eta^H_X) \\
&= \mathcal{P}^+\lambda'_{TX} \cdot \mathcal{P}^+T[\mathrm{id}_{\mathcal{P}^+TX} + \mathcal{P}^+\eta^H_X \cdot \eta^+_X] \cdot \mathcal{P}^+T(h_X + \mathrm{id}_X) \cdot \mathcal{P}^+\mathsf{can} \\
&\cdot \mathcal{P}^+(\kappa_{GX} + \eta^H_X) \\
&= \mathcal{P}^+\lambda'_{TX} \cdot \mathcal{P}^+T[h_X + \eta^+_{TX} \cdot \eta^H_X] \cdot \mathcal{P}^+\mathsf{can} \cdot \mathcal{P}^+(\kappa_{GX} + \eta^H_X) \\
&\qquad \text{(by naturality of } \eta^+)
\end{aligned}
$$

So finally we have shown that the diagrams (8.5) and (8.6) are equivalent. Thus for every set X there is a map h_X that makes the X-component of diagram (8.5) commute.

(2) The maps h_X form a natural transformation. From part (1) together with Theorem 8.23 and Remark 8.24 we know that for every set X we can choose $h_X = e^\dagger_X$ where $e_X = \mathcal{P}^+\mathsf{can}\cdot\mathcal{P}^+(\kappa_{GX}+\eta^H_X)\cdot d_X$. Recall from the proof of Theorem 8.23 that $e^\dagger_X(z) = \{\bar{e}^\dagger_X(\bar{x}) \mid \bar{e}_X \text{ over } e_X \text{ via } f \text{ and } f(\bar{x}) = z\}$. Now let $g : X \to Y$ be any map. To prove that the desired square

$$\begin{array}{ccc} GX & \xrightarrow{e^\dagger_X} & \mathcal{P}^+TX \\ {\scriptstyle Gg}\downarrow & & \downarrow{\scriptstyle \mathcal{P}^+Tg} \\ GY & \xrightarrow[e^\dagger_Y]{} & \mathcal{P}^+TY \end{array}$$

commutes, we consider it pointwise and show the equality

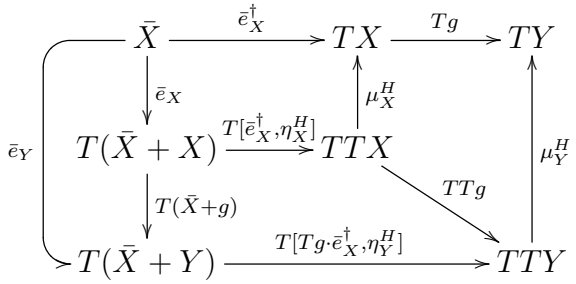
$$(\mathcal{P}^+Tg \cdot e^\dagger_X)(z) = (e^\dagger_Y \cdot Gg)(z)$$

of sets for every $z \in GX$. Consider the two inclusions separately:

$\subseteq$ Assume $t \in (\mathcal{P}^+Tg \cdot e^\dagger_X)(z)$. Then by the definition of $e^\dagger_X$ there exists $\bar{e}_X : \bar{X} \to T(\bar{X} + X)$ over e_X via $f_X : \bar{X} \to GX$ with the unique solution $\bar{e}^\dagger_X : \bar{X} \to TX$ such that $t = (Tg \cdot \bar{e}^\dagger_X)(\bar{x})$ and $f_X(\bar{x}) = z$. We can form $\bar{e}_Y = T(\mathrm{id}_{\bar{X}} + g) \cdot \bar{e}_X : \bar{X} \to T(\bar{X} + Y)$; this is over e_Y via $f_Y = Gg \cdot f_X : \bar{X} \to GY$ since for every $\bar{x} \in \bar{X}$ we have

$$\begin{aligned} & (T(f_Y + \mathrm{id}_Y) \cdot \bar{e}_Y)(\bar{x}) \\ = {} & (T(Gg \cdot f_X + \mathrm{id}_Y) \cdot \bar{e}_Y)(\bar{x}) \\ = {} & (T(Gg \cdot f_X + \mathrm{id}_Y) \cdot T(\mathrm{id}_{\bar{X}} + g) \cdot \bar{e}_X)(\bar{x}) \\ = {} & T(Gg + g) \cdot T(f_X + \mathrm{id}_X) \cdot \bar{e}_X)(\bar{x}) \\ \in {} & (\mathcal{P}^+T(Gg + g) \cdot e_X \cdot f_X)(\bar{x}) && \text{(since } \bar{e}_X \text{ is over } e_X \text{ via } f_X\text{)} \\ = {} & (e_Y \cdot Gg \cdot f_X)(\bar{x}) && \text{(by naturality of } e\text{)} \\ = {} & (e_Y \cdot f_Y)(\bar{x})\,. \end{aligned}$$

We have the unique solution $\bar{e}^\dagger_Y : \bar{X} \to TY$ of $\bar{e}_Y$ for which we prove $\bar{e}^\dagger_Y = Tg \cdot \bar{e}^\dagger_X$ using the following diagram:



(8.7)

The left-hand part is the definition of $\bar{e}_Y$, the upper square is the solution diagram for $\bar{e}_X$, the right-hand part commutes due to naturality of μ^H, and for the lower part remove T and consider the coproduct components separately: the left-hand component trivially commutes, and the right-hand one commutes due to naturality of η^H. Thus the outside commutes, proving $Tg \cdot \bar{e}_X^\dagger$ to be a solution of $\bar{e}_Y$; since this solution is unique, we have $\bar{e}_Y^\dagger = Tg \cdot \bar{e}_X^\dagger$. From this we conclude

$$t = (Tg \cdot \bar{e}_X^\dagger)(\bar{x}) = \bar{e}_Y^\dagger(\bar{x})\,,$$

and notice that $f_Y(\bar{x}) = (Gg \cdot f_X)(\bar{x}) = Gg(z)$. This means $t \in e_Y^\dagger(Gg(z))$ which is $t \in (e_Y^\dagger \cdot Gg)(z)$ as desired.

$\supseteq$ Assume $t \in (e_Y^\dagger \cdot Gg)(z)$. This means there exists $\bar{e}_Y : \bar{Y} \to T(\bar{Y} + Y)$ over e_Y via $f_Y : \bar{Y} \to GY$ with the unique solution $\bar{e}_Y^\dagger : \bar{Y} \to TY$ such that $f_Y(\bar{y}) = Gg(z)$ and $t = \bar{e}_Y^\dagger(\bar{y})$. We want to define a suitable $\bar{e}_X : \bar{X} \to T(\bar{X} + X)$ over e_X via some $f_X : \bar{X} \to GX$. We start by taking the pullback of f_Y and Gg:

$$\begin{array}{ccc} \bar{X} & \overset{f_X}{\dashrightarrow} & GX \\ {\scriptstyle \bar{g}}\,\vdots & & \downarrow{\scriptstyle Gg} \\ \bar{Y} & \xrightarrow[f_Y]{} & GY \end{array} \qquad (8.8)$$

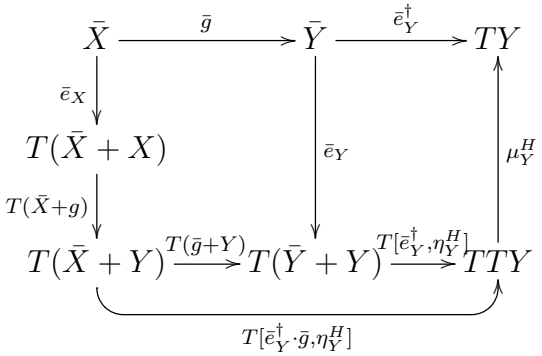
We define $\bar{x} \in \bar{X}$ as the unique element with $\bar{g}(\bar{x}) = \bar{y}$ and $f_X(\bar{x}) = z$—this exists uniquely since $f_Y(\bar{y}) = Gg(z)$.
Now we define a map $\bar{e}'_X : \bar{X} \to T(GX + X)$ as follows: for every $w \in \bar{X}$ we choose as $\bar{e}'_X(w)$ an element $u \in (e_X \cdot f_X)(w)$ with $T(Gg + g)(u) = (T(f_Y + Y) \cdot \bar{e}_Y \cdot \bar{g})(w)$. This is well-defined since such elements always exist: since $\bar{e}_Y$ is over e_Y via f_Y, we know

$$\begin{aligned} (T(f_Y + Y) \cdot \bar{e}_Y \cdot \bar{g})(w) &\in (e_Y \cdot f_Y \cdot \bar{g})(w) \\ &= (e_Y \cdot Gg \cdot f_X)(w) \\ &= (\mathcal{P}^+T(Gg + g) \cdot e_X \cdot f_X)(w)\,. \end{aligned}$$

So there clearly is an element $u \in (e_X \cdot f_X)(w)$ that is mapped by $T(Gg + g)$ to the element $(T(f_Y + Y) \cdot \bar{e}_Y \cdot \bar{g})(w)$. According to our definition of $\bar{e}'_X$, the outside of the diagram

$$\begin{array}{ccccc} & & \overset{\bar{e}'_X}{\frown} & & \\ \bar{X} & \overset{\bar{e}_X}{\dashrightarrow} & T(\bar{X} + X) & \xrightarrow[T(f_X + X)]{} & T(GX + X) \\ {\scriptstyle \bar{g}}\downarrow & & \downarrow{\scriptstyle T(\bar{g} + g)} & & \downarrow{\scriptstyle T(Gg + g)} \\ \bar{Y} & \xrightarrow[\bar{e}_Y]{} & T(\bar{Y} + Y) & \xrightarrow[T(f_Y + Y)]{} & T(GY + Y) \end{array}$$

commutes. Observe that the right-hand square is a pullback: indeed, diagram (8.8) is one, and its coproduct in Set with the pullback $g \cdot \mathrm{id}_X = \mathrm{id}_Y \cdot g$ is a pullback again which is preserved by the functor T. Thus $(\bar{X}, \bar{e}'_X, \bar{e}_Y \cdot \bar{g})$ becomes a competitor for the pullback $(T(\bar{X}+X), T(f_X+X), T(\bar{g}+g))$ and we define $\bar{e}_X : \bar{X} \to T(\bar{X}+X)$ to be the unique mediating map as shown in the diagram. By the definitions of $\bar{e}_X$ and $\bar{e}'_X$ it is easy to show that $\bar{e}_X$ is over e_X via f_X: for every $w \in \bar{X}$ we have $(T(f_X+X) \cdot \bar{e}_X)(w) = \bar{e}'_X(w) \in (e_X \cdot f_X)(w)$. Next we show $Tg \cdot \bar{e}^\dagger_X = \bar{e}^\dagger_Y \cdot \bar{g}$. First recall that $Tg \cdot \bar{e}^\dagger_X$ is the unique solution of $T(\bar{X}+g) \cdot \bar{e}_X$ as shown in diagram (8.7). We show that also $\bar{e}^\dagger_Y \cdot \bar{g}$ is this unique solution and thus is equal to $Tg \cdot \bar{e}^\dagger_X$. To this end, consider the diagram



The right-hand square is the solution diagram for $\bar{e}_Y$, the left-hand square commutes by the definition of $\bar{e}_X$, and the lower part is trivial. Thus the outside commutes showing $\bar{e}^\dagger_Y \cdot \bar{g}$ to be a solution of $T(\bar{X}+g) \cdot \bar{e}_X$ as desired. Now we see that $(Tg \cdot \bar{e}^\dagger_X)(\bar{x}) = (\bar{e}^\dagger_Y \cdot \bar{g})(\bar{x}) = \bar{e}^\dagger_Y(\bar{y}) = t$. Since $f_X(\bar{x}) = z$ this means $t \in (\mathcal{P}^+Tg \cdot e^\dagger_X)(z)$ as desired. □

Recall the partial orders and the term "cutting of a tree" from above Lemma 8.26.

Lemma 8.34. *The $\bar{\mathcal{H}}$-coalgebra homomorphisms $h : G \to \mathcal{P}^+T$ into the weakly final $\bar{\mathcal{H}}$-coalgebra from the proof of Theorem 8.33 are (componentwise) the greatest such homomorphisms; moreover, for every $\bar{\mathcal{H}}$-coalgebra homomorphism $\alpha : G \to \mathcal{P}^+T$ the sets of all finite cuttings of trees from $\alpha_X(z)$ and $h_X(z)$ are the same for every set X and every $z \in GX$.*

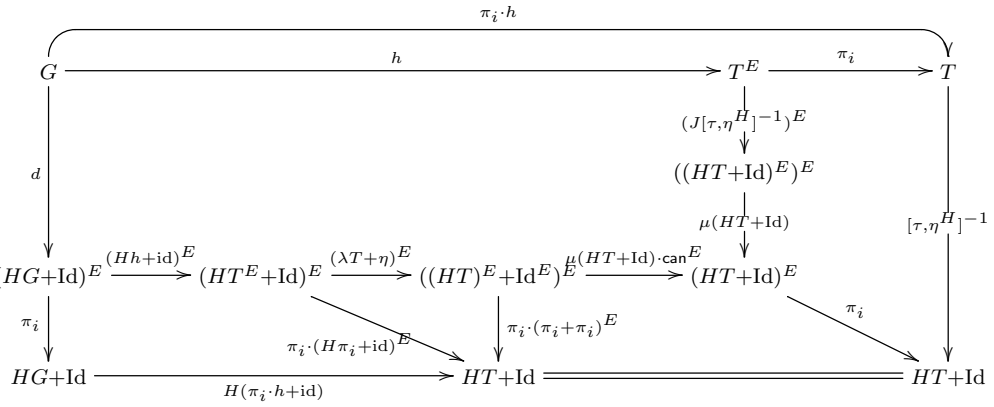
Proof. This follows from Lemma 8.26 and the proof of Theorem 8.33. More detailed, according to part (1) of the proof of Theorem 8.33 the components of $\bar{\mathcal{H}}$-coalgebra homomorphisms into the weakly final coalgebra from this theorem equivalently are solutions of equation morphisms from Theorem 8.23 or Remark 8.24. From the choice of the h_X in part (2) of the proof of Theorem 8.33 we see that these are the greatest solutions as proved in Lemma 8.26. The same lemma also asserts the second part of the statement. □

Final Coalgebras for $\bar{\mathcal{H}}$ in case $M = (-)^E$

Let $((-)^E, \eta, \mu)$ denote the environment monad. Recall again from [MM06], Theorem 5.1 that—provided H is iteratable—$[\tau, \eta^H]^{-1} : T \to HT + \mathrm{Id} = \mathcal{H}T$ is the final $\mathcal{H}$-coalgebra.

Theorem 8.35. *Let λ be the canonical distributive law of any iteratable set functor H over the environment monad $M = (-)^E$ from Example 6.23(2). Then $T \xrightarrow{J[\tau,\eta^H]^{-1}} \bar{\mathcal{H}}T$ is the final $\bar{\mathcal{H}}$-coalgebra.*

Proof. Given any $\bar{\mathcal{H}}$-coalgebra $d : G \to (HG + \mathrm{Id})^E$ we prove that there is a unique homomorphism $G \to T^E$ into the $\bar{\mathcal{H}}$-coalgebra $J[\tau, \eta^H]^{-1} : T \to (HT + \mathrm{Id})^E$, i. e. we prove that the upper left-hand square in the diagram



in $[\mathsf{Set}, \mathsf{Set}]$ commutes. To this end we show that h is an $\bar{\mathcal{H}}$-coalgebra homomorphism (the upper left-hand square commutes) precisely if $\pi_i \cdot h$ is an $\mathcal{H}$-coalgebra homomorphism for every $i \in E$ (the outside commutes for every $i \in E$). Then it follows from finality of $[\tau, \eta^H]^{-1}$ for the functor $\mathcal{H}$ that $J[\tau, \eta^H]^{-1}$ is final for the functor $\bar{\mathcal{H}}$.

In fact, all inner parts of the above diagram except the upper left-hand part commute: the upper part is trivial, for the lower left-hand part we use naturality of π_i, and for the triangle use that we have $\pi_i \cdot \lambda = H\pi_i$ and $\pi_i \cdot \eta = \mathrm{id}$ by the definitions of λ and η (see Example 6.23(2) and Example 6.1(5)). For the lower right-hand part use that $\pi_i \cdot \mu = \pi_i \cdot \pi_i^E$ by the definition of μ (see Example 6.1(5)), and the remaining right-hand part commutes by

$$\mu(HT+\mathrm{Id})\cdot(J[\tau,\eta^H]^{-1})^E = \mu(HT+\mathrm{Id})\cdot(\eta(HT+\mathrm{Id})\cdot[\tau,\eta^H]^{-1})^E = ([\tau,\eta^H]^{-1})^E$$

and naturality of π_i. Thus the upper left-hand square commutes precisely if the outside commutes for every $i \in E$. □

For the special case of $E = 1$, i. e. $M = \mathrm{Id}$, Theorem 8.35 states that for the canonical distributive law $\lambda = \mathrm{id}$ of any iteratable set endofunctor H over M (see Example 6.23(2)), $J[\tau, \eta^H]^{-1} = [\tau, \eta^H]^{-1}$ is the final coalgebra for $\bar{\mathcal{H}} = \mathcal{H}$; thus we get back Theorem 5.1 from [MM06].

8.4 Monad Morphisms

In this section, we establish that certain coalgebra homomorphisms into the (weakly) final $\bar{\mathcal{H}}$-coalgebras from Section 8.3 are monad morphisms. This is important for proving recursive program schemes with effects to have unique or—in case of $\mathcal{P}^+$—canonical greatest solutions (Section 8.5). The proofs for the different cases according to the monads M are similar, so we start with the most complicated case which is the nonempty powerset monad $M = \mathcal{P}^+$ since here we have no final but only weakly final coalgebras. For all other cases we then explain how the proof can be simplified and adapted.

To simplify notation, for the rest of Chapter 8 we use the following notational convention:

Notation 8.36. F denotes the free monad F^{H+V} on $H + V$. For all other free monads or free CIMs we explicitly add the functor on which they are free as in F^H and T^H. We assume that M is one of the monads $\mathrm{Id} + 1$, $\mathcal{P}$, $\mathcal{D}$, $\mathcal{P}^+$ or $(-)^E$ from Example 6.1. By $\lambda : HM \to MH$ we denote from now on a canonical distributive law (see Theorem 6.19 and Example 6.23, (1)–(2)) and by $\lambda' : F^H M \to M F^H$ or $\lambda' : T^H M \to M T^H$ its extension (see Propositions 8.2 and 8.12 and Corollary 8.7). Whenever we write $\bar{\mathcal{H}}$, we mean the functor induced by a canonical distributive law λ (see Lemma 8.29 and below this lemma).

Monad Morphisms in case $M = \mathcal{P}^+$

Definition 8.37. Given a natural transformation $e' : V \to \mathcal{P}^+ HF$ where H and V are finitary polynomial endofunctors on Set, we define the following $\bar{\mathcal{H}}$-coalgebra:

$$p = (\, F \xrightarrow{[\phi^{H+V}, \eta^{H+V}]^{-1}} (H+V)F + \mathrm{Id} \xrightarrow{[\eta^+ HF \cdot H\eta^{H+V}, e']F + \eta^+} \mathcal{P}^+ HFF + \mathcal{P}^+ \xrightarrow{\mathcal{P}^+ H\mu^{H+V} + \mathrm{id}} \mathcal{P}^+ HF + \mathcal{P}^+ \xrightarrow{\mathsf{can}} \mathcal{P}^+(HF + \mathrm{Id})\,) \,. \tag{8.9}$$

By Theorem 8.33 there is a (componentwise greatest) natural transformation h such that the diagram

$$\begin{array}{ccc} F & \xrightarrow{h} & T^H \\ {\scriptstyle p}\downarrow & & \downarrow{\scriptstyle J[\tau^H,\eta^H]^{-1}} \\ HF + \mathrm{Id} & \xrightarrow[\tilde{\mathcal{H}}h]{} & HT^H + \mathrm{Id} \end{array} \tag{8.10}$$

commutes (in $[\mathsf{Set},\mathsf{Set}]_{\mathcal{M}}$). Observe that diagram (8.10) translates to

$$h = \mathcal{P}^+[\tau^H,\eta^H]\cdot\mu^+(HT^H+\mathrm{Id})\cdot\mathcal{P}^+\mathsf{can}\cdot\mathcal{P}^+(\lambda T^H+\eta^+)\cdot\mathcal{P}^+(Hh+\mathrm{id})\cdot p \tag{8.11}$$

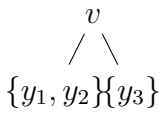
in $[\mathsf{Set},\mathsf{Set}]$.

Remarks 8.38.

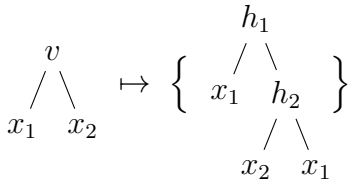
1. We give a concrete description of the natural transformation h: let Σ and Γ be the signatures associated with H and V. Given a tree from FY, h_Y substitutes level by level all subtrees headed by operation symbols from Γ by the sets of trees given by e', and all subtrees headed by operation symbols from Σ or leaves labeled by a variable from Y by the singleton set containing this subtree or variable. Then it continues the same way for all child trees of all trees inside the substituted sets. Although the original tree is finite, this might be an infinite process since for every level there might be a set plugged in the tree which contains again a tree having an operation symbol from Γ. The resulting set from $\mathcal{P}^+T^HY$ contains all those (possibly infinite) trees that are built by starting at the root and choosing an element whenever a set is hit.

2. Comparing our definition of h with the approach in [MM06] (see the natural transformation h defined in the proof of Theorem 6.5), the definition of h in loc. cit. is given in two steps, first by a second-order substitution once over the whole tree and then repeating this in the definition of h. This causes problems in the case of nondeterministic natural transformations e' since we did not assume a distributive law of V over $\mathcal{P}^+$; even if we took the canonical one (see Example 6.14(4)), applying it before substitutions in subtrees are performed, one may "loose" trees, see the following Example 8.39. So the order in which substitutions and nondeterministic choices are made matters. However, one can use second-order substitution also here in case e' only involves finite sets: then we can regard the nondeterministic choice "or" as a given operation first and use repeated second-order substitution; in a second step we interpret the or making it into a two-element set everywhere and using λ and μ^+ afterwards repeatedly.

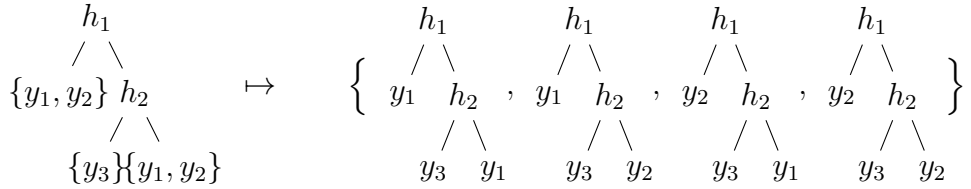
Example 8.39. If we defined h as in [MM06], during the levelwise execution of h we may encounter trees with sets of trees at a level below some operation symbol $v \in \Gamma$, for example the following tree:



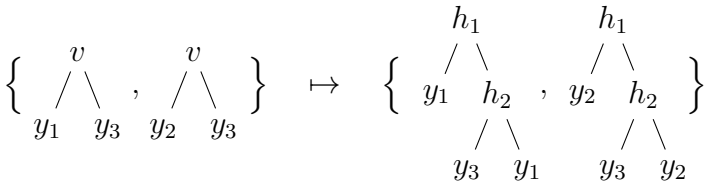
Now consider the (part of a) natural transformation $e'_X : VX \to \mathcal{P}^+HFX$ given by the assignment



where $v \in \Gamma$ and $h_1, h_2 \in \Sigma$ are binary operations. If we apply this first to the above tree from $F\mathcal{P}^+Y$, we obtain the left-hand tree



from $T^H\mathcal{P}^+Y$; and the repeated application of the canonical distributive law λ (which is an application of λ') gives the right-hand set from $\mathcal{P}^+T^HY$. On the other hand, applying first the canonical distributive law of V over $\mathcal{P}^+$ (which acts the same way as λ), we obtain the left-hand set



from $\mathcal{P}^+FY$; but the following substitution according to e' only results in a two-element subset of the four-element set of trees above.

Proposition 8.40. *The natural transformation $h : F \to \mathcal{P}^+T^H$ from Definition 8.37 is a monad morphism.*

Proof. We prove that h is a monad morphism between the free monad $(F, \eta^{H+V}, \mu^{H+V})$ and the composite monad $(\mathcal{P}^+T^H, \eta^+ * \eta^H, (\mu^+ * \mu^H) \cdot \mathcal{P}^+\lambda' T^H)$.

(1) We start with the first monad morphism law $h \cdot \eta^{H+V} = \eta^+ * \eta^H$ for h. First we see that

$$\begin{aligned} p \cdot \eta^{H+V} &= \mathsf{can} \cdot (\mathcal{P}^+ H\mu^{H+V} + \mathrm{id}) \cdot ([\eta^+ HF \cdot H\eta^{H+V}, e']F + \eta^+) \\ &\quad \cdot [\phi^{H+V}, \eta^{H+V}]^{-1} \cdot \eta^{H+V} && \text{(by (8.9))} \\ &= \mathsf{can} \cdot (\mathcal{P}^+ H\mu^{H+V} + \mathrm{id}) \cdot ([\eta^+ HF \cdot H\eta^{H+V}, e']F + \eta^+) \cdot \mathsf{inr} \\ &= \mathcal{P}^+\mathsf{inr} \cdot \eta^+ \end{aligned}$$

and then have

$$\begin{aligned} h \cdot \eta^{H+V} &= \mathcal{P}^+[\tau^H, \eta^H] \cdot \mu^+(HT^H + \mathrm{Id}) \cdot \mathcal{P}^+\mathsf{can} \cdot \mathcal{P}^+(\lambda T^H + \eta^+) \\ &\quad \cdot \mathcal{P}^+(Hh + \mathrm{id}) \cdot p \cdot \eta^{H+V} && \text{(by (8.11))} \\ &= \mathcal{P}^+[\tau^H, \eta^H] \cdot \mu^+(HT^H + \mathrm{Id}) \cdot \mathcal{P}^+\mathsf{can} \cdot \mathcal{P}^+(\lambda T^H + \eta^+) \\ &\quad \cdot \mathcal{P}^+(Hh + \mathrm{id}) \cdot \mathcal{P}^+\mathsf{inr} \cdot \eta^+ && \text{(by the above equation)} \\ &= \mathcal{P}^+[\tau^H, \eta^H] \cdot \mu^+(HT^H + \mathrm{Id}) \cdot \mathcal{P}^+\mathcal{P}^+\mathsf{inr} \cdot \mathcal{P}^+\eta^+ \cdot \eta^+ \\ &= \mathcal{P}^+\eta^H \cdot \mu^+ \cdot \mathcal{P}^+\eta^+ \cdot \eta^+ \\ &= \mathcal{P}^+\eta^H \cdot \eta^+ && \text{(by one of the unit laws for } \mathcal{P}^+) \\ &= \eta^+ * \eta^H \,. \end{aligned} \tag{8.12}$$

(2) We prepare the proof of the second monad morphism law by calculating

$$\begin{aligned} p \cdot \phi^{H+V} &= \mathsf{can} \cdot (\mathcal{P}^+ H\mu^{H+V} + \mathrm{id}) \cdot ([\eta^+ HF \cdot H\eta^{H+V}, e']F + \eta^+) \\ &\quad \cdot [\phi^{H+V}, \eta^{H+V}]^{-1} \cdot \phi^{H+V} && \text{(by (8.9))} \\ &= \mathsf{can} \cdot (\mathcal{P}^+ H\mu^{H+V} + \mathrm{id}) \cdot ([\eta^+ HF \cdot H\eta^{H+V}, e']F + \eta^+) \cdot \mathsf{inl} \\ &= \mathcal{P}^+\mathsf{inl} \cdot \mathcal{P}^+ H\mu^{H+V} \cdot [\eta^+ HF \cdot H\eta^{H+V}, e']F \end{aligned}$$

and then

$$\begin{aligned}
h \cdot \phi^{H+V} &= \mathcal{P}^+[\tau^H, \eta^H] \cdot \mu^+(HT^H + \mathrm{Id}) \cdot \mathcal{P}^+\mathsf{can} \cdot \mathcal{P}^+(\lambda T^H + \eta^+) \\
&\quad \cdot \mathcal{P}^+(Hh + \mathrm{id}) \cdot p \cdot \phi^{H+V} \qquad \text{(by (8.11))} \\
&= \mathcal{P}^+[\tau^H, \eta^H] \cdot \mu^+(HT^H + \mathrm{Id}) \cdot \mathcal{P}^+\mathsf{can} \cdot \mathcal{P}^+(\lambda T^H + \eta^+) \\
&\quad \cdot \mathcal{P}^+(Hh + \mathrm{id}) \cdot \mathcal{P}^+\mathsf{inl} \cdot \mathcal{P}^+ H\mu^{H+V} \\
&\quad \cdot [\eta^+ HF \cdot H\eta^{H+V}, e']F \qquad \text{(by the above equation)} \qquad (8.13) \\
&= \mathcal{P}^+[\tau^H, \eta^H] \cdot \mu^+(HT^H + \mathrm{Id}) \cdot \mathcal{P}^+\mathcal{P}^+\mathsf{inl} \cdot \mathcal{P}^+\lambda T^H \\
&\quad \cdot \mathcal{P}^+ Hh \cdot \mathcal{P}^+ H\mu^{H+V} \cdot [\eta^+ HF \cdot H\eta^{H+V}, e']F \\
&= \mathcal{P}^+\tau^H \cdot \mu^+ HT^H \cdot \mathcal{P}^+\lambda T^H \cdot \mathcal{P}^+ Hh \cdot \mathcal{P}^+ H\mu^{H+V} \\
&\quad \cdot [\eta^+ HF \cdot H\eta^{H+V}, e']F\,.
\end{aligned}$$

(3) We prove the second monad morphism law $h \cdot \mu^{H+V} = (\mu^+ * \mu^H) \cdot \mathcal{P}^+\lambda' T^H \cdot (h * h)$ for h. Consider the $\bar{\mathcal{H}}$-coalgebra $q : FF \to \mathcal{P}^+(HFF + \mathrm{Id})$ given by

$$\begin{array}{c}
FF \\
\downarrow{\scriptstyle [\phi^{H+V}, \eta^{H+V}]^{-1}F} \\
((H+V)F + \mathrm{Id})F = \\
(H+V)FF + F \\
\downarrow{\scriptstyle (H+V)FF + [\phi^{H+V}, \eta^{H+V}]^{-1}} \\
(H+V)FF + (H+V)F + \mathrm{Id} \\
\downarrow{\scriptstyle [\eta^+ HF \cdot H\eta^{H+V}, e']FF + [\eta^+ HF \cdot H\eta^{H+V}, e']F + \eta^+} \\
\mathcal{P}^+ HFFF + \mathcal{P}^+ HFF + \mathcal{P}^+ \\
\downarrow{\scriptstyle \mathcal{P}^+ H\mu^{H+V}F + \mathcal{P}^+ H\mu^{H+V} + \mathcal{P}^+} \\
\mathcal{P}^+ HFF + \mathcal{P}^+ HF + \mathcal{P}^+ \\
\downarrow{\scriptstyle \mathcal{P}^+ HFF + \mathcal{P}^+\mathsf{can}} \\
\mathcal{P}^+ HFF + \mathcal{P}^+(HF + \mathrm{Id}) \\
\downarrow{\scriptstyle [\mathcal{P}^+\mathsf{inl}, \mathcal{P}^+(H\eta^{H+V}F + \mathrm{id})]} \\
\mathcal{P}^+(HFF + \mathrm{Id})\,.
\end{array}$$

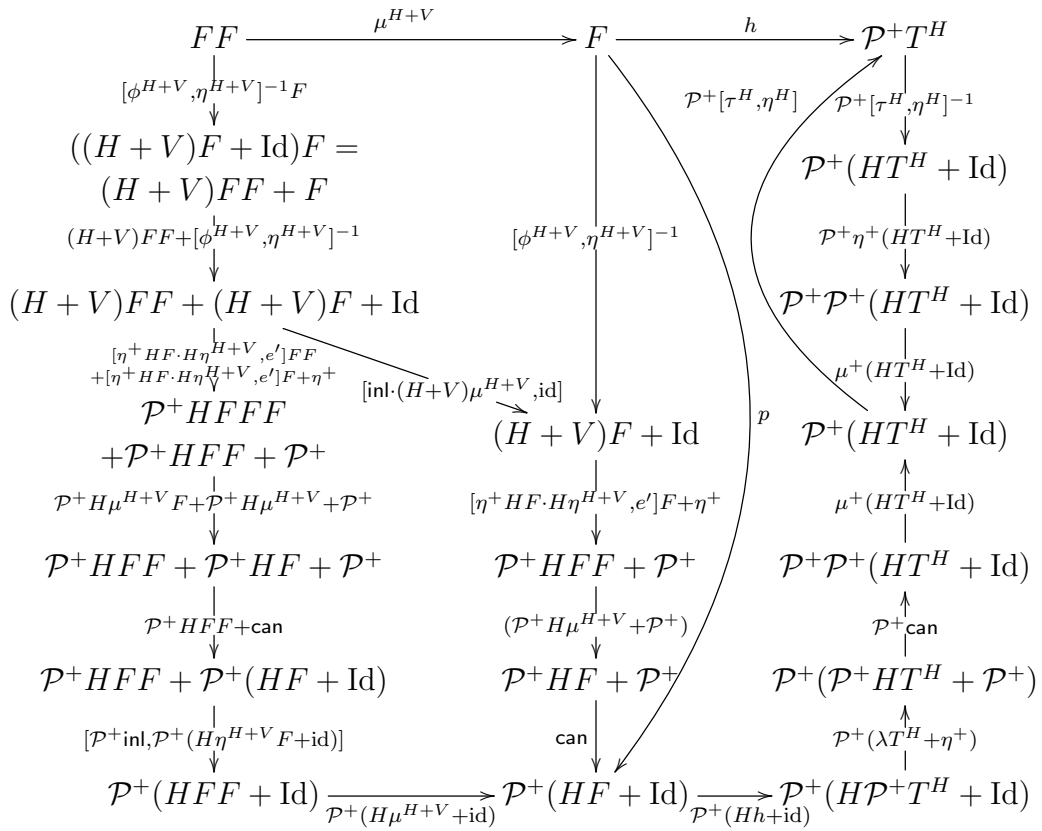
We show that both sides of

$$h \cdot \mu^{H+V} = (\mu^+ * \mu^H) \cdot \mathcal{P}^+\lambda' T^H \cdot (h * h) \qquad (8.14)$$

are the componentwise greatest $\bar{\mathcal{H}}$-coalgebra homomorphism from q into the

weakly final coalgebra $J[\tau^H, \eta^H]^{-1}$, see Theorem 8.33 and Lemma 8.34, and thus prove the second monad morphism law.

(a) The left-hand side of (8.14) is a homomorphism. We need to show commutativity of the following diagram:



The small upper right-hand part uses one of the monad unit laws and then reverses the isomorphism $[\tau^H, \eta^H]$; the squeezed right-hand part between the two bent arrows is the definition of h, see (8.11). The diagram on the other side of the big bent arrow p is the definition of p, see (8.9); for the upper left-hand part reverse all isomorphisms $[\phi^{H+V}, \eta^{H+V}]^{-1}$ and consider the three coproduct components of $(H+V)FF+(H+V)F+\mathrm{Id}$ separately: for the left-hand component we see that $\mu^{H+V}\cdot\phi^{H+V}F = \phi^{H+V}\cdot(H+V)\mu^{H+V}$ (see Remark 2.23), the middle component commutes since $\mu^{H+V}\cdot\eta^{H+V}F\cdot\phi^{H+V} = \phi^{H+V}$ and for the right-hand component we have $\mu^{H+V}\cdot\eta^{H+V}F\cdot\eta^{H+V} = \eta^{H+V}$. For the remaining lower left-hand part of the above diagram we again consider the coproduct components of $(H+V)FF+(H+V)F+\mathrm{Id}$ separately: for the left-hand component we have

$$\begin{aligned}&\mathcal{P}^+H\mu^{H+V}\cdot\mathcal{P}^+H\mu^{H+V}F\cdot[\eta^+HF\cdot H\eta^{H+V},e']FF\\ =\;&\mathcal{P}^+H\mu^{H+V}\cdot[\eta^+HF\cdot H\eta^{H+V},e']F\cdot(H+V)\mu^{H+V}\end{aligned}$$

due to the multiplication law for F and naturality of $[\eta^+ HF \cdot H\eta^{H+V}, e']$, and for the middle and right-hand component taken together we see that the diagram commutes since $\mathcal{P}^+(H\mu^{H+V} + \mathrm{id}) \cdot \mathcal{P}^+(H\eta^{H+V}F + \mathrm{id}) = \mathrm{id}$. Thus $h \cdot \mu^{H+V}$ is an $\bar{\mathcal{H}}$-coalgebra homomorphism between q and the final coalgebra.

(b) Next we show that the left-hand side of (8.14) is the componentwise greatest such homomorphism. We already know from part (1) the proof of Theorem 8.33 that the component $h_X \cdot \mu_X^{H+V}$ of the $\bar{\mathcal{H}}$-coalgebra homomorphism into the final coalgebra equivalently is a solution of the equation morphism $\mathfrak{q}_X = \mathcal{P}^+\mathsf{can} \cdot \mathcal{P}^+(\kappa^H_{FFX} + \eta^H_X) \cdot q_X$. Let $\mathfrak{q}^\dagger_X : FFX \to \mathcal{P}^+T^HX$ be the greatest solution of $\mathfrak{q}_X$, then we show $h_X \cdot \mu_X^{H+V} \geq \mathfrak{q}^\dagger_X$ elementwise, i. e. we show

$$(h_X \cdot \mu_X^{H+V})(v) \supseteq \mathfrak{q}^\dagger_X(v)$$

for every $v \in FFX$. To this end let $t \in \mathfrak{q}^\dagger_X(v)$, i. e. by Lemma 8.26 and (8.4) we have $\bar{\mathfrak{q}}_X : \bar{X} \to T^H(\bar{X} + X)$ over $\mathfrak{q}_X$ via $f : \bar{X} \to FFX$ such that $f(\bar{x}) = v$ and $t = \bar{\mathfrak{q}}^\dagger_X(\bar{x})$. It is not difficult to see that $\bar{\mathfrak{q}}_X$ is also over $\mathfrak{p}_X = \mathcal{P}^+\mathsf{can} \cdot \mathcal{P}^+(\kappa^H_{FX} + \eta^H_X) \cdot p_X$ but this time via $\mu_X^{H+V} \cdot f : \bar{X} \to FX$: first we see that $p_X \cdot \mu_X^{H+V} = \mathcal{P}^+(H\mu_X^{H+V} + X) \cdot q_X$ (the left-hand half of the diagram from part (a) of the proof) implies, together with naturality of κ^H and can, commutativity of the following diagram:

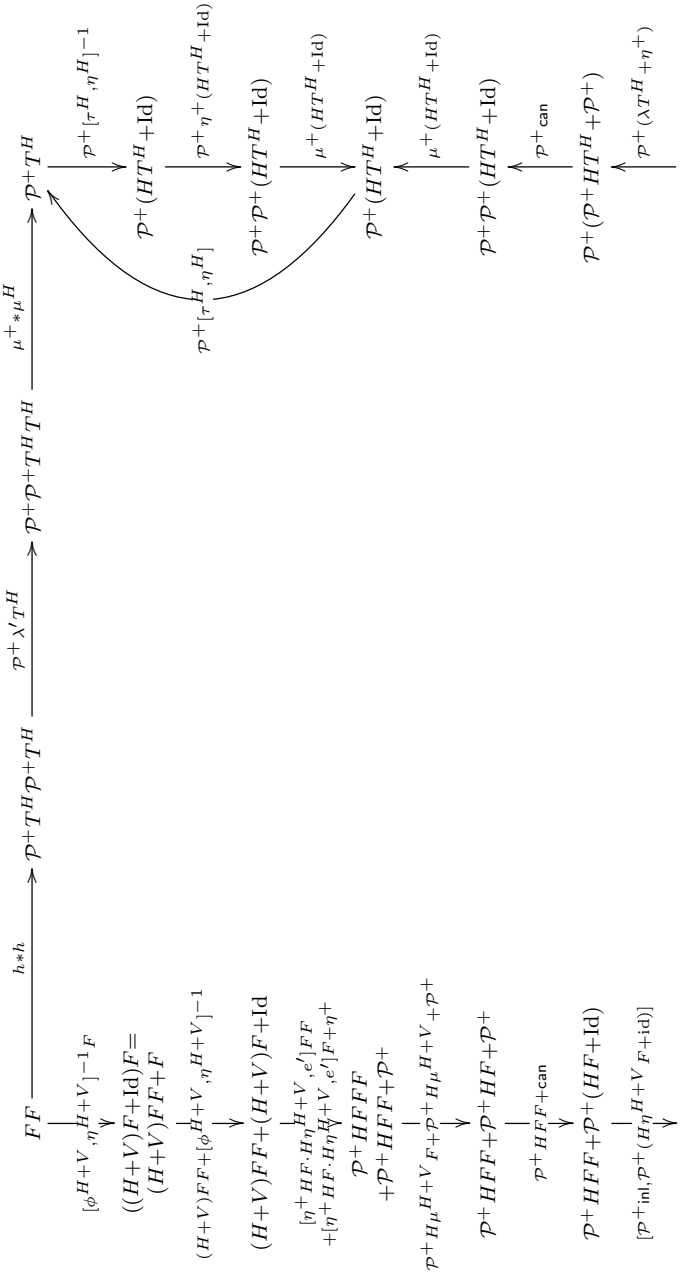
$$\begin{array}{ccccc}
 & FFX & \xrightarrow{\mu^{H+V}} & FX & \\
 & {\scriptstyle q_X}\downarrow & & \downarrow{\scriptstyle p_X} & \\
 & \mathcal{P}^+(HFFX + X) & \xrightarrow{\mathcal{P}^+(H\mu_X^{H+V} + X)} & \mathcal{P}^+(HFX + X) & \\
{\scriptstyle \mathfrak{q}_X} & {\scriptstyle \mathcal{P}^+(\kappa^H_{FFX} + \eta^H_X)}\downarrow & & \downarrow{\scriptstyle \mathcal{P}^+(\kappa^H_{FX} + \eta^H_X)} & {\scriptstyle \mathfrak{p}_X} \\
 & \mathcal{P}^+(T^HFFX + T^HX) & \xrightarrow{\mathcal{P}^+(T^H\mu_X^{H+V} + T^HX)} & \mathcal{P}^+(T^HFX + T^HX) & \\
 & {\scriptstyle \mathcal{P}^+\mathsf{can}}\downarrow & & \downarrow{\scriptstyle \mathcal{P}^+\mathsf{can}} & \\
 & \mathcal{P}^+T^H(FFX + X) & \xrightarrow{\mathcal{P}^+T^H(\mu_X^{H+V} + X)} & \mathcal{P}^+T^H(FX + X) &
\end{array} \tag{8.15}$$

Using this, we have

$$\begin{aligned}
& (T^H(\mu_X^{H+V} \cdot f + X) \cdot \bar{\mathfrak{q}}_X)(\bar{x}) \\
= {} & (T^H(\mu_X^{H+V} + X) \cdot T^H(f + X) \cdot \bar{\mathfrak{q}}_X)(\bar{x}) \\
\in {} & (\mathcal{P}^+T^H(\mu_X^{H+V} + X) \cdot \mathfrak{q}_X \cdot f)(\bar{x}) && \text{(since } \bar{\mathfrak{q}}_X \text{ is over } \mathfrak{q}_X \text{ via } f\text{)} \\
= {} & (\mathfrak{p}_X \cdot \mu_X^{H+V} \cdot f)(\bar{x}) && \text{(by diagram (8.15))}
\end{aligned}$$

for every $\bar{x} \in \bar{X}$. From $f(\bar{x}) = v$ we get $(\mu_X^{H+V} \cdot f)(\bar{x}) = \mu_X^{H+V}(v)$ and thus $t = \bar{\mathfrak{q}}^\dagger_X(\bar{x}) \in h_X(\mu_X^{H+V}(v)) = (h_X \cdot \mu_X^{H+V})(v)$ as desired since h_X is the greatest solution of $\mathfrak{p}_X$ and thus is characterized by (8.4).

(c) We show that the right-hand side of (8.14) is a coalgebra homomorphism between the same coalgebras q and $J[\tau^H, \eta^H]^{-1}$ as the left-hand side. We need to show that the diagram



commutes. To this end, we use the same small upper right-hand part as in the diagram from part (a) of the proof and reverse the isomorphisms $[\phi^{H+V}, \eta^{H+V}]^{-1}$ (the first two downwards arrows on the left-hand side). Then

we consider the three coproduct components of $(H+V)FF+(H+V)F+\mathrm{Id}$ separately. For the left-hand component we first establish

$$
\begin{aligned}
&\mathcal{P}^+\tau^H\cdot\mu^+HT^H\cdot\mathcal{P}^+\lambda T^H\cdot\mathcal{P}^+H(\mu^+*\mu^H)\cdot\mathcal{P}^+H\mathcal{P}^+\lambda' T^H\\
&=\mu^+T^H\cdot\mathcal{P}^+\mathcal{P}^+\tau^H\cdot\mathcal{P}^+\lambda T^H\cdot\mathcal{P}^+H\mu^+T^H\cdot\mathcal{P}^+H\mathcal{P}^+\mathcal{P}^+\mu^H\cdot\mathcal{P}^+H\mathcal{P}^+\lambda' T^H\\
&\qquad\qquad\text{(by naturality of }\mu^+)\\
&=\mu^+T^H\cdot\mathcal{P}^+\mathcal{P}^+\tau^H\cdot\mathcal{P}^+\mu^+HT^H\cdot\mathcal{P}^+\mathcal{P}^+\lambda T^H\cdot\mathcal{P}^+\lambda\mathcal{P}^+T^H\cdot\mathcal{P}^+H\mathcal{P}^+\mathcal{P}^+\mu^H\\
&\quad\cdot\mathcal{P}^+H\mathcal{P}^+\lambda' T^H\qquad\qquad\text{(by the axiom for }\lambda\text{ and }\mu^+)\\
&=\mu^+T^H\cdot\mathcal{P}^+\mathcal{P}^+\tau^H\cdot\mathcal{P}^+\mathcal{P}^+H\mu^H\cdot\mathcal{P}^+\mu^+HT^HT^H\cdot\mathcal{P}^+\mathcal{P}^+\lambda T^HT^H\\
&\quad\cdot\mathcal{P}^+\lambda\mathcal{P}^+T^HT^H\cdot\mathcal{P}^+H\mathcal{P}^+\lambda' T^H\qquad\qquad\text{(by naturality of }\lambda\text{ and }\mu^+)\\
&=\mu^+T^H\cdot\mathcal{P}^+\mathcal{P}^+\mu^H\cdot\mathcal{P}^+\mathcal{P}^+\tau^HT^H\cdot\mathcal{P}^+\mu^+HT^HT^H\cdot\mathcal{P}^+\mathcal{P}^+\lambda T^HT^H\\
&\quad\cdot\mathcal{P}^+\mathcal{P}^+H\lambda' T^H\cdot\mathcal{P}^+\lambda T^H\mathcal{P}^+T^H\qquad\text{(by Remark 2.48 and naturality of }\lambda)\\
&=(\mu^+*\mu^H)\cdot\mathcal{P}^+\mu^+T^HT^H\cdot\mathcal{P}^+\mathcal{P}^+\mathcal{P}^+\tau^HT^H\cdot\mathcal{P}^+\mathcal{P}^+\lambda T^HT^H\\
&\quad\cdot\mathcal{P}^+\mathcal{P}^+H\lambda' T^H\cdot\mathcal{P}^+\lambda T^H\mathcal{P}^+T^H\qquad\qquad\text{(by naturality of }\mu^+)\\
&=(\mu^+*\mu^H)\cdot\mu^+\mathcal{P}^+T^HT^H\cdot\mathcal{P}^+\mathcal{P}^+\lambda' T^H\cdot\mathcal{P}^+\mathcal{P}^+\tau^H\mathcal{P}^+T^H\cdot\mathcal{P}^+\lambda T^H\mathcal{P}^+T^H\\
&\qquad\qquad\text{(by the multiplication law for }\mathcal{P}^+\text{ and (8.2) with }M=\mathcal{P}^+)\\
&=(\mu^+*\mu^H)\cdot\mathcal{P}^+\lambda' T^H\cdot\mathcal{P}^+\tau^H\mathcal{P}^+T^H\cdot\mu^+HT^H\mathcal{P}^+T^H\cdot\mathcal{P}^+\lambda T^H\mathcal{P}^+T^H\\
&\qquad\qquad\text{(by naturality of }\mu^+).
\end{aligned}
$$

This yields the desired equality for the left-hand component:

$$
\begin{aligned}
&\mathcal{P}^+\tau^H\cdot\mu^+HT^H\cdot\mathcal{P}^+\lambda T^H\cdot\mathcal{P}^+H(\mu^+*\mu^H)\cdot\mathcal{P}^+H\mathcal{P}^+\lambda' T^H\cdot\mathcal{P}^+H(h*h)\\
&\quad\cdot\mathcal{P}^+H\mu^{H+V}F\cdot[\eta^+HF\cdot H\eta^{H+V},e']FF\\
&=(\mu^+*\mu^H)\cdot\mathcal{P}^+\lambda' T^H\cdot\mathcal{P}^+\tau^H\mathcal{P}^+T^H\cdot\mu^+HT^H\mathcal{P}^+T^H\cdot\mathcal{P}^+\lambda T^H\mathcal{P}^+T^H\\
&\quad\cdot\mathcal{P}^+H(h*h)\cdot\mathcal{P}^+H\mu^{H+V}F\cdot[\eta^+HF\cdot H\eta^{H+V},e']FF\\
&\qquad\qquad\text{(by the above equation)}\\
&=(\mu^+*\mu^H)\cdot\mathcal{P}^+\lambda' T^H\cdot(h*h)\cdot\phi^{H+V}F\qquad\qquad\text{(by (8.13))}
\end{aligned}
$$

For the middle component we have

$$\begin{aligned}
&\mathcal{P}^+\tau^H \cdot \mu^+ HT^H \cdot \mathcal{P}^+\lambda T^H \cdot \mathcal{P}^+ H(\mu^+ * \mu^H) \cdot \mathcal{P}^+ H\mathcal{P}^+\lambda' T^H \cdot \mathcal{P}^+ H(h * h) \\
&\cdot \mathcal{P}^+ H\eta^{H+V} F \cdot \mathcal{P}^+ H\mu^{H+V} \cdot [\eta^+ HF \cdot H\eta^{H+V}, e']F \\
=\ &\mathcal{P}^+\tau^H \cdot \mu^+ HT^H \cdot \mathcal{P}^+\lambda T^H \cdot \mathcal{P}^+ H(\mu^+ * \mu^H) \cdot \mathcal{P}^+ H\mathcal{P}^+\lambda' T^H \\
&\cdot \mathcal{P}^+ H\eta^+ T^H \mathcal{P}^+ T^H \cdot \mathcal{P}^+ H\eta^H \mathcal{P}^+ T^H \cdot \mathcal{P}^+ Hh \cdot \mathcal{P}^+ H\mu^{H+V} \\
&\cdot [\eta^+ HF \cdot H\eta^{H+V}, e']F && \text{(by (8.12))} \\
=\ &\mathcal{P}^+\tau^H \cdot \mu^+ HT^H \cdot \mathcal{P}^+\lambda T^H \cdot \mathcal{P}^+ H\mathcal{P}^+\mu^H \cdot \mathcal{P}^+ H\lambda' T^H \cdot \mathcal{P}^+ H\eta^H \mathcal{P}^+ T^H \\
&\cdot \mathcal{P}^+ Hh \cdot \mathcal{P}^+ H\mu^{H+V} \cdot [\eta^+ HF \cdot H\eta^{H+V}, e']F \\
&&& \text{(by naturality of } \eta^+ \text{ and one of the unit laws for } \mathcal{P}^+) \\
=\ &\mathcal{P}^+\tau^H \cdot \mu^+ HT^H \cdot \mathcal{P}^+\lambda T^H \cdot \mathcal{P}^+ H\mathcal{P}^+\mu^H \cdot \mathcal{P}^+ H\mathcal{P}^+\eta^H T^H \cdot \mathcal{P}^+ Hh \\
&\cdot \mathcal{P}^+ H\mu^{H+V} \cdot [\eta^+ HF \cdot H\eta^{H+V}, e']F && \text{(by the axiom for } \lambda' \text{ and } \eta^+) \\
=\ &\mathcal{P}^+\tau^H \cdot \mu^+ HT^H \cdot \mathcal{P}^+\lambda T^H \cdot \mathcal{P}^+ Hh \cdot \mathcal{P}^+ H\mu^{H+V} \cdot [\eta^+ HF \cdot H\eta^{H+V}, e']F \\
&&& \text{(by one of the unit laws for } T^H) \\
=\ &h \cdot \phi^{H+V} && \text{(by (8.13))} \\
=\ &\mathcal{P}^+\mu^H \cdot \mathcal{P}^+\eta^H T^H \cdot h \cdot \phi^{H+V} && \text{(by one of the unit laws for } T^H) \\
=\ &\mathcal{P}^+\mu^H \cdot \lambda' T^H \cdot \eta^H \mathcal{P}^+ T^H \cdot h \cdot \phi^{H+V} && \text{(by the axiom for } \lambda' \text{ and } \eta^H) \\
=\ &(\mu^+ * \mu^H) \cdot \mathcal{P}^+\lambda' T^H \cdot \eta^+ T^H \mathcal{P}^+ T^H \cdot \eta^H \mathcal{P}^+ T^H \cdot h \cdot \phi^{H+V} \\
&&& \text{(by one of the unit laws for } \mathcal{P}^+ \text{ and naturality of } \eta^+) \\
=\ &(\mu^+ * \mu^H) \cdot \mathcal{P}^+\lambda' T^H \cdot (h * h) \cdot \eta^{H+V} F \cdot \phi^{H+V} && \text{(by (8.12))}.
\end{aligned}$$

For the right-hand component we have

$$\begin{aligned}
&\mathcal{P}^+\eta^H \cdot \mu^+ \cdot \mathcal{P}^+\eta^+ \cdot \eta^+ \\
=\ &\mathcal{P}^+\eta^H \cdot \eta^+ && \text{(by one of the unit laws for } \mathcal{P}^+) \\
=\ &\eta^+ T^H \cdot \eta^H && \text{(by naturality of } \eta^+) \\
=\ &\mathcal{P}^+\mu^H \cdot \mathcal{P}^+\eta^H T^H \cdot h \cdot \eta^{H+V} && \text{(by (8.12) and one of the unit laws for } T^H) \\
=\ &\mathcal{P}^+\mu^H \cdot \lambda' T^H \cdot \eta^H \mathcal{P}^+ T^H \cdot h \cdot \eta^{H+V} && \text{(by the axiom for } \lambda' \text{ and } \eta^H) \\
=\ &(\mu^+ * \mu^H) \cdot \mathcal{P}^+\lambda' T^H \cdot \eta^+ T^H \mathcal{P}^+ T^H \cdot \eta^H \mathcal{P}^+ T^H \cdot h \cdot \eta^{H+V} \\
&&& \text{(by one of the unit laws for } \mathcal{P}^+ \text{ and naturality of } \eta^+) \\
=\ &(\mu^+ * \mu^H) \cdot \mathcal{P}^+\lambda' T^H \cdot (h * h) \cdot \eta^{H+V} F \cdot \eta^{H+V} \\
&&& \text{(by (8.12) and naturality of } \eta^{H+V}).
\end{aligned}$$

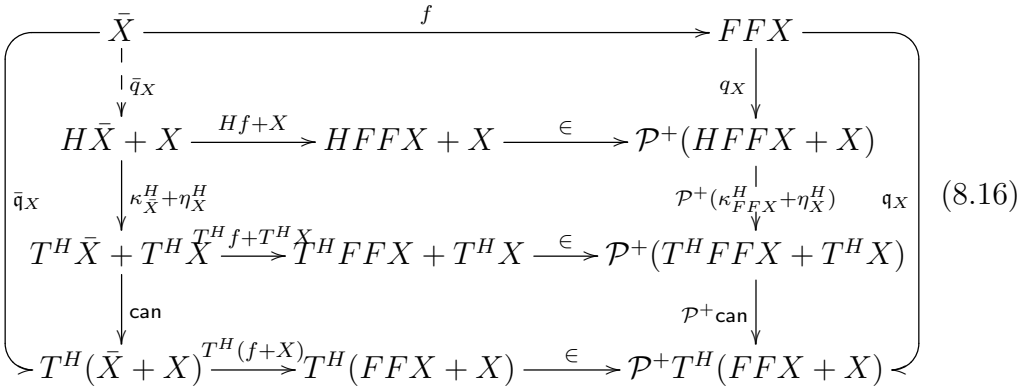
Thus the above diagram commutes and $(\mu^+ * \mu^H) \cdot \mathcal{P}^+\lambda' T^H \cdot (h * h)$ is an $\bar{\mathcal{H}}$-coalgebra homomorphism.

(d) Finally we prove that the right-hand side of (8.14) is the componentwise greatest homomorphism. We already know from part (1) of the proof of Theorem 8.33 that the component $(\mu^+ * \mu^H)_X \cdot \mathcal{P}^+\lambda'_{T^H X} \cdot (h * h)_X$ of the $\bar{\mathcal{H}}$-coalgebra homomorphism into the final coalgebra equivalently is a solution of the equation morphism $\mathfrak{q}_X = \mathcal{P}^+\mathsf{can} \cdot \mathcal{P}^+(\kappa^H_{FFX} + \eta^H_X) \cdot q_X$. Let $\mathfrak{q}^\dagger_X : FFX \to \mathcal{P}^+T^HY$ be the greatest solution of $\mathfrak{q}_X$, then we show $(\mu^+ * \mu^H)_X \cdot \mathcal{P}^+\lambda'_{T^H X} \cdot (h * h)_X \supseteq \mathfrak{q}^\dagger_X$ elementwise, i. e. we show

$$((\mu^+ * \mu^H)_X \cdot \mathcal{P}^+\lambda'_{T^H X} \cdot (h * h)_X)(v) \supseteq \mathfrak{q}^\dagger_X(v)$$

for every $v \in FFX$. To this end let $t \in \mathfrak{q}^\dagger_X(v)$, i. e. by Lemma 8.26 and (8.4) we have $\bar{\mathfrak{q}}_X : \bar{X} \to T^H(\bar{X}+X)$ over $\mathfrak{q}_X$ via $f : \bar{X} \to FFX$ such that $f(\bar{x}) = v$ and $t = \bar{\mathfrak{q}}^\dagger_X(\bar{x})$.

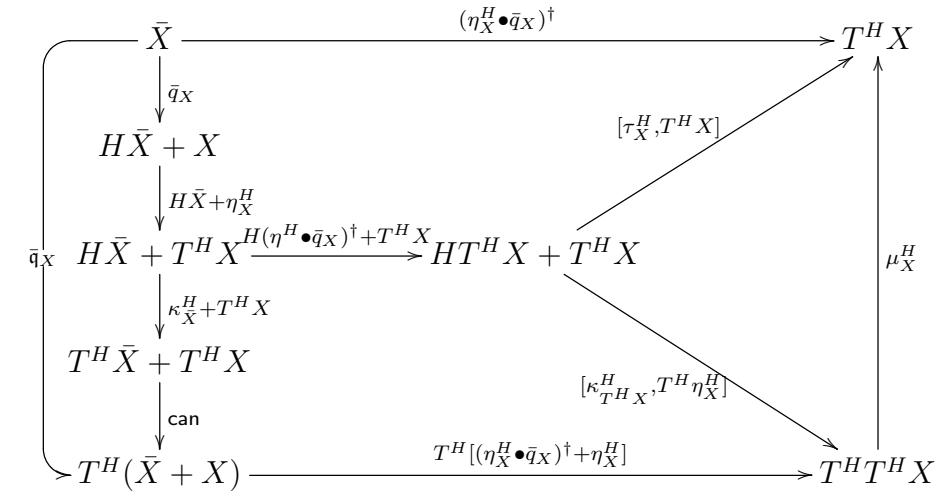
We start by observing that the fact that the equation morphism $\bar{\mathfrak{q}}_X$ is over $\mathfrak{q}_X$ breaks down, along the structure of $\mathfrak{q}_X$, to the fact that $\bar{\mathfrak{q}}_X = \mathsf{can} \cdot (\kappa^H_{\bar{X}} + \eta^H_X) \cdot \bar{q}_X$ where $\bar{q}_X : \bar{X} \to H\bar{X} + X$ is a flat equation morphism such that $((Hf+X)\cdot\bar{q}_X)(\bar{x}) \in (q_X \cdot f)(\bar{x})$ for all $\bar{x} \in \bar{X}$: consider the diagram



(8.16)

to see this, where the arrows labeled by $\in$ stand for the membership relation: the outside "commutes" since $\bar{\mathfrak{q}}_X$ is over $\mathfrak{q}_X$ and the right-hand part commutes by the definition of $\mathfrak{q}_X$. Since the center and lower squares also "commute", we see that there is $\bar{q}_X$ as desired.

Next recall Notation 2.36 and consider the following diagram:



(8.17)

All inner parts commute: the left-hand part commutes by the definition of $\bar{q}_X$ (see diagram (8.16)), the upper part since $(\eta^H_X \bullet \bar{q}_X)^\dagger$ is the unique solution of $\eta^H_X \bullet \bar{q}_X$ in the (free) CIA τ^H_X, for the lower part use naturality of κ^H, and for the right-hand part use Lemma 2.49 and one of the unit laws for the monad T^H. Thus the outside commutes showing $\bar{\mathfrak{q}}^\dagger_X = (\eta^H_X \bullet \bar{q}_X)^\dagger$ for the unique solution of $\bar{\mathfrak{q}}_X$.

Next we split $\bar{X} = \bar{X}_0 + \bar{X}_1$ such that $([\phi^{H+V}, \eta^{H+V}]^{-1}_{FX} \cdot f)[\bar{X}_0] \subseteq (H + V)FFX$ and $([\phi^{H+V}, \eta^{H+V}]^{-1}_{FX} \cdot f)[\bar{X}_1] \subseteq FX$. Observe that by the definition of q_X in the beginning of part (3) of the proof (see the $\eta^{H+V}F$ in the end) q_X has a restriction to the right-hand component of $FFX \cong (H+V)FFX+FX$ via $\eta^{H+V}F$; moreover, this restriction is $p_X : FX \to \mathcal{P}^+(HFX + X)$ from Definition 8.37, i. e. $q_X \cdot \eta^{H+V}_{FX} = \mathcal{P}^+(H\eta^{H+V}_{FX} + X) \cdot p_X$. Thus also $\bar{q}_X$ restricts to $\bar{X}_1$ via $\mathsf{inr} : \bar{X}_1 \to \bar{X}$, i. e. we have $\bar{p}_X : \bar{X}_1 \to H\bar{X}_1 + X$ such that

$$(H\mathsf{inr} + X) \cdot \bar{p}_X = \bar{q}_X \cdot \mathsf{inr}\,; \tag{8.18}$$

moreover, we have $((Hf_1 + X) \cdot \bar{p}_X)(\bar{x}) \in (p_X \cdot f_1)(\bar{x})$ for all $\bar{x} \in \bar{X}_1$ where $f_1 : \bar{X}_1 \to FX$ is the restriction of f to $\bar{X}_1$, i. e. $\eta^{H+V}_{FX} \cdot f_1 = f \cdot \mathsf{inr}$.

We form $\bar{\mathfrak{p}}_X = \mathsf{can} \cdot (\kappa^H_{\bar{X}_1} + \eta^H_X) \cdot \bar{p}_X$ and use a diagram similar to (8.16) to see that $\bar{\mathfrak{p}}_X$ is over $\mathfrak{p}_X$ via f_1. Also a diagram similar to (8.17) shows $\bar{\mathfrak{p}}^\dagger_X = (\eta^H_X \bullet \bar{p}_X)^\dagger$.

Observe from the definition of q_X that it restricts to a map $o_{FX} : (H + V)FFX \to \mathcal{P}^+HFFX$ when we consider only the left-hand component of $FFX \cong (H+V)FFX+FX$; moreover, $o : (H+V)F \to \mathcal{P}^+HF$ is a natural transformation. Thus also $\bar{q}_X$ restricts to a map $\bar{o}_{FX} : \bar{X}_0 \to H\bar{X}$ when we consider only $\bar{X}_0$, i. e.

$$\mathsf{inl} \cdot \bar{o}_{FX} = \bar{q}_X \cdot \mathsf{inl}\,; \tag{8.19}$$

and we have $(Hf \cdot \bar{o}_{FX})(\bar{x}) \in (o_X \cdot f_0)(\bar{x})$ for all $\bar{x} \in \bar{X}_0$, where $f_0 : \bar{X}_0 \to (H+V)FFX$ is the restriction of f to $\bar{X}_0$, i. e. $\phi_{FX}^{H+V} \cdot f_0 = f \cdot \mathsf{inl}$.

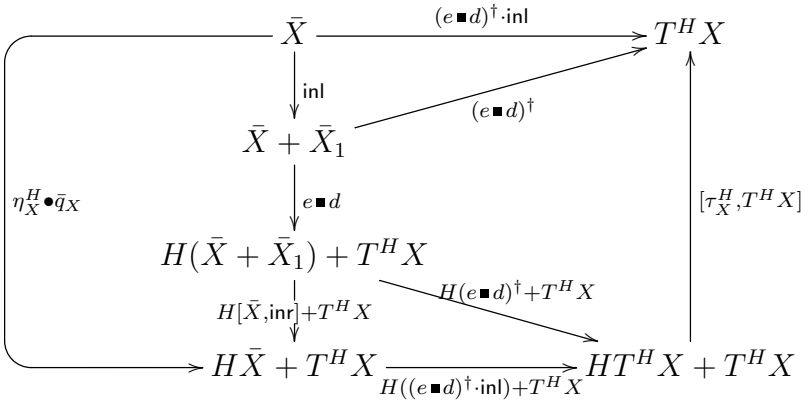
In order to use the compositionality property of flat equation morphisms w. r. t. solutions in CIAs (see Remark 4.9) we form the following flat equation morphisms

$$d = (\bar{X} = \bar{X}_0 + \bar{X}_1 \xrightarrow{\bar{o}_{FX}+\bar{X}_1} H\bar{X} + \bar{X}_1) \quad \text{and} \quad e = (\bar{X}_1 \xrightarrow{\eta_X^H \bullet \bar{p}_X} H\bar{X}_1 + T^H X)\,.$$

The composite $e \blacksquare d$ is the map

$$\begin{array}{ccc} \bar{X} + \bar{X}_1 & \xrightarrow{[(\bar{o}_{FX}+\bar{X}_1),\mathsf{inr}]} & H\bar{X} + \bar{X}_1 \\ & & \downarrow{\scriptstyle H\bar{X}+(\eta_X^H \bullet \bar{p}_X)} \\ & & H\bar{X} + H\bar{X}_1 + T^H X \xrightarrow{\mathsf{can}+T^H X} H(\bar{X} + \bar{X}_1) + T^H X \end{array}$$

and we immediately see that $(e \blacksquare d) \cdot \mathsf{inr} = (e \blacksquare d) \cdot \mathsf{inl} \cdot \mathsf{inr}$; let $(e \blacksquare d)^\dagger$ be the unique solution of $e \blacksquare d$ in the CIA τ_X^H, i. e. $(e \blacksquare d)^\dagger = [\tau_X^H, T^H X] \cdot (H(e \blacksquare d)^\dagger + T^H X) \cdot (e \blacksquare d)$. Consequently, we have $(e \blacksquare d)^\dagger \cdot \mathsf{inr} = (e \blacksquare d)^\dagger \cdot \mathsf{inl} \cdot \mathsf{inr}$ which makes the lower triangle of the following diagram commute:



The upper triangle trivially commutes, the right-hand part commutes since $(e \blacksquare d)^\dagger$ is the unique solution of $e \blacksquare d$, and for the remaining left-hand part we

calculate

$$\begin{aligned}
\eta^H_X \bullet \bar{q}_X &= (H\bar{X} + \eta^H_X) \cdot [\mathsf{inl} \cdot \bar{o}_{FX}, (H\mathsf{inr} + X) \cdot \bar{p}_X] \\
&\quad \text{(by the definition of } \bullet \text{, by (8.18) and (8.19))} \\
&= (H\bar{X} + \eta^H_X) \cdot [\mathsf{inl}, H\mathsf{inr} + X] \cdot (\bar{o}_{FX} + \bar{p}_X) \\
&= (H\bar{X} + \eta^H_X) \cdot (H[\bar{X}, \mathsf{inr}] + X) \cdot (\mathsf{can} + X) \cdot (\bar{o}_{FX} + \bar{p}_X) \\
&= (H[\bar{X}, \mathsf{inr}] + T^H X) \cdot (\mathsf{can} + T^H X) \cdot (H\bar{X} + (\eta^H_X \bullet \bar{p}_X)) \\
&\quad \cdot (\bar{o}_{FX} + \bar{X}_1) \\
&= (H[\bar{X}, \mathsf{inr}] + T^H X) \cdot (e \blacksquare d) \cdot \mathsf{inl} \\
&\quad \text{(by the definition of } e \blacksquare d).
\end{aligned}$$

Thus the outside commutes showing

$$(\eta^H_X \bullet \bar{q}_X)^\dagger = (e \blacksquare d)^\dagger \cdot \mathsf{inl} \tag{8.20}$$

by the uniqueness of solutions of flat equation morphisms in the (free) CIA τ^H_X. But by the compositionality property in the same CIA we also know that $(e \blacksquare d)^\dagger \cdot \mathsf{inl} = (e^\dagger \bullet d)^\dagger$, where the right-hand side is (by the definition of d and e and the above equality for $\bar{\mathfrak{p}}^\dagger_X$)

$$((\eta^H_X \bullet \bar{p}_X)^\dagger \bullet (\bar{o}_{FX} + \bar{X}_1))^\dagger = (\bar{\mathfrak{p}}^\dagger_X \bullet (\bar{o}_{FX} + \bar{X}_1))^\dagger = (\bar{o}_{FX} + \bar{\mathfrak{p}}^\dagger_X)^\dagger . \tag{8.21}$$

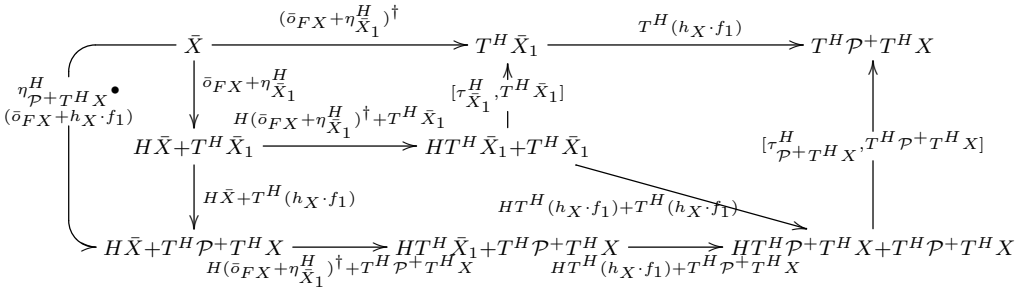
Since $\bar{\mathfrak{p}}_X$ is over $\mathfrak{p}_X$ via f_1, and we know $\mathfrak{p}^\dagger_X = h_X$ for the greatest solution of $\mathfrak{p}_X$ by the definition of h_X (Definition 8.37) and part (1) of the proof of Theorem 8.33, we have $\bar{\mathfrak{p}}^\dagger_X(\bar{x}) \in (h_X \cdot f_1)(\bar{x})$ for all $\bar{x} \in \bar{X}_1$, cf. (8.4). Thus, the flat equation morphism $\bar{o}_{FX} + \bar{\mathfrak{p}}^\dagger_X : \bar{X} \to H\bar{X} + T^H X$ is over $\bar{o}_{FX} + h_X \cdot f_1 : \bar{X} \to H\bar{X} + \mathcal{P}^+ T^H X$ via $\mathrm{id}_{\bar{X}}$ (cf. Definition 7.30) which implies

$$(\bar{o}_{FX} + \bar{\mathfrak{p}}^\dagger_X)^\dagger(\bar{x}) \in (\bar{o}_{FX} + h_X \cdot f_1)^\dagger(\bar{x}) \tag{8.22}$$

for every $\bar{x} \in \bar{X}$, where we have the greatest solution in the H-algebra $\mathcal{P}^+ \tau^H_X \cdot \lambda_{T^H X}$ on the right-hand side (cf. Definition 7.31 and Proposition 7.33). Applying Remark 8.14, we see that

$$\begin{aligned}
(\bar{o}_{FX} + h_X \cdot f_1)^\dagger &= \mathcal{P}^+ \mu^H_X \cdot \lambda'_{T^H X} \cdot T^H(h_X \cdot f_1) \cdot (\bar{o}_{FX} + \eta^H_{\bar{X}_1})^\dagger \\
&= \mathcal{P}^+ \mu^H_X \cdot \lambda'_{T^H X} \cdot (\eta^H_{\mathcal{P}^+ T^H X} \bullet (\bar{o}_{FX} + h_X \cdot f_1))^\dagger ;
\end{aligned} \tag{8.23}$$

for the last equation consider the following commutative diagram:



The left-hand part commutes by the definition of $\bullet$, the upper left-hand square since $(\bar{o}_{FX}+\eta^H_{\bar{X}_1})^\dagger$ is the (unique) solution of $\bar{o}_{FX}+\eta^H_{\bar{X}_1}$, the right-hand part commutes due to naturality of τ^H and the lower part is trivial. Thus the outside commutes and uniqueness of solutions in the (free) CIA $\tau^H_{\mathcal{P}^+T^HX}$ yields $T^H(h_X\cdot f_1)\cdot(\bar{o}_{FX}+\eta^H_{\bar{X}_1})^\dagger=(\eta^H_{\mathcal{P}^+T^HX}\bullet(\bar{o}_{FX}+h_X\cdot f_1))^\dagger$ as desired.

Let us denote the morphism $\bar{o}_{FX}+h_X\cdot f_1:\bar{X}\to H\bar{X}+\mathcal{P}^+T^HX$ by $\bar{r}_X$. We show that for all $\bar{x}\in\bar{X}$ we have

$$((H(Fh_X\cdot f)+\mathcal{P}^+T^HX)\cdot\bar{r}_X)(\bar{x})\in(p_{\mathcal{P}^+T^HX}\cdot Fh_X\cdot f)(\bar{x})\,.$$

To this end, consider the coproduct components of $\bar{X}=\bar{X}_0+\bar{X}_1$ separately: for every $\bar{x}\in\bar{X}_0$, by the above equation for o_{FX} and $\bar{o}_{FX}$ and naturality of o we obtain

$$\begin{aligned}(H(Fh_X\cdot f)\cdot\bar{o}_{FX})(\bar{x})&\in(\mathcal{P}^+HFh_X\cdot o_{FX}\cdot f_0)(\bar{x})\\&=(o_{\mathcal{P}^+T^HX}\cdot(H+V)Fh_X\cdot f_0)(\bar{x})\,;\end{aligned}$$

and for every $\bar{x}\in\bar{X}_1$ we trivially obtain $(h_X\cdot f_1)(\bar{x})\in(\eta^+_{\mathcal{P}^+T^HX}\cdot h_X\cdot f_1)(\bar{x})$ where the right-hand side is indeed as desired since

$$\begin{aligned}&p_{\mathcal{P}^+T^HX}\cdot Fh_X\cdot f\cdot\mathsf{inr}\\&=p_{\mathcal{P}^+T^HX}\cdot Fh_X\cdot\eta^{H+V}_{FX}\cdot f_1&&\text{(by the above equation for } f\text{ and } f_1)\\&=p_{\mathcal{P}^+T^HX}\cdot\eta^{H+V}_{\mathcal{P}^+T^HX}\cdot h_X\cdot f_1&&\text{(by naturality of } \eta^{H+V})\\&=\mathcal{P}^+\mathsf{inr}\cdot\eta^+_{\mathcal{P}^+T^HX}\cdot h_X\cdot f_1&&\text{(see part (1) of the proof).}\end{aligned}$$

We form $\bar{\mathfrak{r}}_X=\mathsf{can}\cdot(\kappa^H_{\bar{X}}+\eta^H_{\mathcal{P}^+T^HX})\cdot\bar{r}_X$ and see that again by a diagram similar to (8.16), $\bar{\mathfrak{r}}_X$ is over $\mathfrak{p}_{\mathcal{P}^+T^HX}$ via $Fh_X\cdot f$. Also a diagram similar to (8.17) shows

$$\bar{\mathfrak{r}}^\dagger_X=(\eta^H_{\mathcal{P}^+T^HX}\bullet\bar{r}_X)^\dagger\,. \tag{8.24}$$

We are now ready to prove the desired result $t \in ((\mu^+ * \mu^H)_X \cdot \mathcal{P}^+\lambda'_{T^H X} \cdot (h * h)_X)(v)$; recall that $t = \bar{\mathfrak{q}}^\dagger_X(\bar{x})$ and $f(\bar{x}) = v$. We start with

$$\begin{aligned}
t &= \bar{\mathfrak{q}}^\dagger_X(\bar{x}) \\
&= (\eta^H_X \bullet \bar{q}_X)^\dagger(\bar{x}) && \text{(by (8.17))} \\
&= ((e \blacksquare d)^\dagger \cdot \mathsf{inl})(\bar{x}) && \text{(by (8.20))} \\
&= (e^\dagger \bullet d)^\dagger(\bar{x}) && \text{(by compositionality of } d \text{ and } e\text{)} \\
&= (\bar{o}_{FX} + \bar{\mathfrak{p}}^\dagger_X)^\dagger(\bar{x}) && \text{(by (8.21))}
\end{aligned}$$

and continue with

$$\begin{aligned}
t &= (\bar{o}_{FX} + \bar{\mathfrak{p}}^\dagger_X)^\dagger(\bar{x}) && \text{(see above)} \\
&\in (\bar{o}_{FX} + h_X \cdot f_1)^\dagger(\bar{x}) && \text{(by (8.22))} \\
&= (\mathcal{P}^+\mu^H_X \cdot \lambda'_{T^H X} \cdot (\eta^H_{\mathcal{P}^+T^H X} \bullet (\bar{o}_{FX} + h_X \cdot f_1))^\dagger)(\bar{x}) && \text{(by (8.23))} \\
&= (\mathcal{P}^+\mu^H_X \cdot \lambda'_{T^H X} \cdot (\eta^H_{\mathcal{P}^+T^H X} \bullet \bar{r}_X)^\dagger)(\bar{x}) && \text{(definition of } \bar{r}_X\text{)} \\
&= (\mathcal{P}^+\mu^H_X \cdot \lambda'_{T^H X} \cdot \bar{\mathfrak{r}}^\dagger_X)(\bar{x}) && \text{(by (8.24)).}
\end{aligned}$$

Finally, since $\bar{\mathfrak{r}}_X$ is over $\mathfrak{p}_{\mathcal{P}^+T^H X}$ via $Fh_X \cdot f$ which implies $\bar{\mathfrak{r}}^\dagger_X(\bar{x}) \in (\mathfrak{p}^\dagger_{\mathcal{P}^+T^H X} \cdot Fh_X \cdot f)(\bar{x})$ for all $\bar{x} \in \bar{X}$ for the greatest solution $\mathfrak{p}^\dagger_{\mathcal{P}^+T^H X}$ of $\mathfrak{p}_{\mathcal{P}^+T^H X}$ (cf. (8.4) and Lemma 8.26), and since we know $\mathfrak{p}^\dagger_{\mathcal{P}^+T^H X} = h_{\mathcal{P}^+T^H X}$ from the definition of h (Definition 8.37) and part (1) of the proof of Theorem 8.33, we obtain

$$\begin{aligned}
t &\in (\mu^+_{T^H X} \cdot \mathcal{P}^+\mathcal{P}^+\mu^H_X \cdot \mathcal{P}^+\lambda'_{T^H X} \cdot \mathfrak{p}^\dagger_{\mathcal{P}^+T^H X} \cdot Fh_X \cdot f)(\bar{x}) \\
&= ((\mu^+ * \mu^H)_X \cdot \mathcal{P}^+\lambda'_{T^H X} \cdot (h * h)_X)(v)
\end{aligned}$$

as desired. □

Monad Morphisms in case $M = (-)^E$

The following definition is a variant of Definition 8.37 for the environment monad $((-)^E, \eta, \mu)$ instead of the nonempty powerset monad $(\mathcal{P}^+, \eta^+, \mu^+)$.

Definition 8.41. Given a natural transformation $e' : V \to (HF)^E$ where H is an iteratable functor, we define an $\bar{\mathcal{H}}$-coalgebra

$$p = (\, F \xrightarrow[[\eta HF \cdot H\eta^{H+V}, e']F + \eta]{[\phi^{H+V}, \eta^{H+V}]^{-1}} (H+V)F + \mathrm{Id}$$

$$(HFF)^E + \mathrm{Id}^E \xrightarrow{(H\mu^{H+V})^E + \mathrm{id}} (HF)^E + \mathrm{Id}^E \xrightarrow{\mathsf{can}} (HF + \mathrm{Id})^E \,).$$

By Theorem 8.35 there is a unique natural transformation h such that the diagram

$$\begin{array}{ccc} F & \xrightarrow{h} & T^H \\ {\scriptstyle p}\downarrow & & \downarrow{\scriptstyle J[\tau^H,\eta^H]^{-1}} \\ HF + \mathrm{Id} & \xrightarrow[\bar{\mathcal{H}}h]{} & HT^H + \mathrm{Id} \end{array} \tag{8.25}$$

commutes (in $[\mathsf{Set}, \mathsf{Set}]_{\mathcal{M}}$).

Proposition 8.42. *The natural transformation $h : F \to (T^H)^E$ from Definition 8.41 is a monad morphism.*

Proof. The proof of Proposition 8.40 can be adapted as follows: the nonempty powerset monad $(\mathcal{P}^+, \eta^+, \mu^+)$ is replaced by the environment monad $((-)^E, \eta, \mu)$ and the parts (3b) and (3d) are omitted. In fact, these parts are no longer needed due to the uniqueness of coalgebra homomorphisms into the final $\bar{\mathcal{H}}$-coalgebra (see Theorem 8.35). Furthermore, replace (8.9) and (8.11) from Definition 8.37 by Definition 8.41, Theorem 8.33 and Lemma 8.34 by Theorem 8.35 and (8.2) with $M = \mathcal{P}^+$ by (8.2) with $M = (-)^E$. □

Corollary 8.43. *The natural transformation $h : F \to T^H$ from Definition 8.41 in case $E = 1$ is a monad morphism.*

Remark 8.44. Since the monad morphisms h from Definition 8.41 in case $E = 1$ have the free monad F as their domain, they are uniquely determined by $h \cdot \kappa^{H+V}$. We will see later in the proof of Theorem 8.63 that $h \cdot \kappa^{H+V} = [\kappa^H, e^\dagger]$ for a natural transformation $e^\dagger : V \to T^H$. Now Corollary 8.43 follows from the more general result from [MM06] that $h' : T^{H+V} \to T^H$, uniquely determined by $h' \cdot \kappa^{H+V} = [\kappa^H, e^\dagger]$ for the same $e^\dagger$, is a monad morphism: we have $h = h' \cdot (\kappa^{H+V})^\#$ where $(\kappa^{H+V})^\# : F \to T^{H+V}$ is the unique monad morphism extending κ^{H+V} (cf. Definition 2.8).

Monad Morphisms under Assumption 7.16

We can also vary Definition 8.37 for the setting from Assumption 7.16 which includes the maybe monad $M = \mathrm{Id} + 1$, the powerset monad $M = \mathcal{P}$ and the subdistribution monad $M = \mathcal{D}$.

Definition 8.45. Let Assumption 7.16 hold true. Given a natural transformation $e' : V \to MHF$, we define an $\bar{\mathcal{H}}$-coalgebra

$$p = (\, F \xrightarrow{[\phi^{H+V},\eta^{H+V}]^{-1}} (H+V)F + \mathrm{Id} \xrightarrow{[\eta^M HF \cdot H\eta^{H+V}, e']F + \eta^M} MHFF + M \xrightarrow{MH\mu^{H+V}+\mathrm{id}} MHF + M \xrightarrow{\mathsf{can}} M(HF + \mathrm{Id})\,) .$$

By Theorem 8.30 there is a unique natural transformation h such that the diagram

$$\begin{array}{ccc} F & \xrightarrow{h} & F^H \\ {\scriptstyle p}\downarrow & & \downarrow{\scriptstyle J[\phi^H,\eta^H]^{-1}} \\ HF + \mathrm{Id} & \xrightarrow[\bar{\mathcal{H}}h]{} & HF^H + \mathrm{Id} \end{array} \tag{8.26}$$

commutes (in $[\mathsf{Set}, \mathsf{Set}]_{\mathcal{M}}$).

Proposition 8.46. *The natural transformation $h : F \to MF^H$ from Definition 8.45 is a monad morphism.*

Proof. The proof of Proposition 8.40 can be adapted as follows: the nonempty powerset monad $(\mathcal{P}^+, \eta^+, \mu^+)$ is replaced by the monad (M, η^M, μ^M) and the parts (3b) and (3d) are omitted. In fact, these parts are no longer needed due to the uniqueness of coalgebra homomorphisms into the final $\bar{\mathcal{H}}$-coalgebra (see Theorem 8.30). Furthermore, replace the free CIM T^H by the free monad F^H, τ^H by ϕ^H and Remark 2.48 by Remark 2.23; also replace (8.9) and (8.11) from Definition 8.37 by Definition 8.45, Theorem 8.33 and Lemma 8.34 by Theorem 8.30 and (8.2) with $M = \mathcal{P}^+$ by (8.1). □

Remark 8.47. The distributive law $\lambda : HM \to MH$ induced by Assumption 7.16 in Definition 8.45 and Proposition 8.46 needs not necessarily be the canonical one from Theorem 6.19. Indeed, different from the results for the monads $\mathcal{P}^+$ (Proposition 8.40) and $(-)^E$ (Proposition 8.42), where we needed to specialize to the canonical ones already in Section 7.2, we never needed to add such a condition to Assumption 7.16.

8.5 Uninterpreted Solutions of RPS's with Effects

This Section contains the main results of Chapter 8: for the five monads from Example 6.1 we define RPS's with the respective effects, introduce

the notion of guarded such RPS's and say what uninterpreted solutions are. We then prove that guarded RPS's with effects have unique uninterpreted solutions (except for the effect of "nonempty nondeterminism" where we obtain canonical uninterpreted solutions). However, one has to say that we restrict the solution space to MF^H when we look at partial, nondeterministic and probabilistic RPS's ($M = \mathrm{Id} + 1$, $M = \mathcal{P}$ and $M = \mathcal{D}$); in case of nonempty nondeterministic and composite RPS's ($M = \mathcal{P}^+$ and $M = (-)^E$) we consider solutions in MT^H. The proofs make use of the technical results from Sections 8.1 to 8.4.

Recall Notation 8.36 from Section 8.4.

8.5.1 Partial, Nondeterministic and Probabilistic RPS's

Definition 8.48. Let H and V be analytic functors and let (M, η^M, μ^M) be a commutative monad. A *recursive program scheme with M-effects* (or *M-RPS*, for short) is a natural transformation $e : V \to MF$. It is called *guarded* if it factors as follows:

$$e \equiv (V \xrightarrow{e'} MHF \xrightarrow{M\mathsf{inl}F} M(H+V)F \xrightarrow{M\phi^{H+V}} MF)\,.$$

An *uninterpreted solution* of e is a natural transformation $e^\dagger : V \to MF^H$ such that the diagram

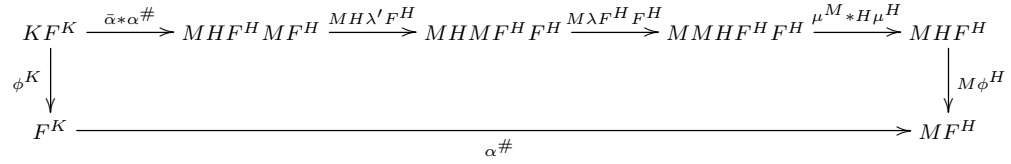
$$\begin{array}{ccc} V & \xrightarrow{\quad e^\dagger \quad} & MF^H \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle \mu^M F^H} \\ MF & \xrightarrow[M[\eta^M F^H \cdot \kappa^H, e^\dagger]^\#]{} & MMF^H \end{array} \tag{8.27}$$

commutes. Here $[\eta^M F^H \cdot \kappa^H, e^\dagger]^\#$ is the unique monad morphism into MF^H extending $[\eta^M F^H \cdot \kappa^H, e^\dagger]$ (see Definition 2.8), where the composite monad MF^H is induced by the extension (see Proposition 8.2) of the canonical distributive law (see Theorem 6.19).

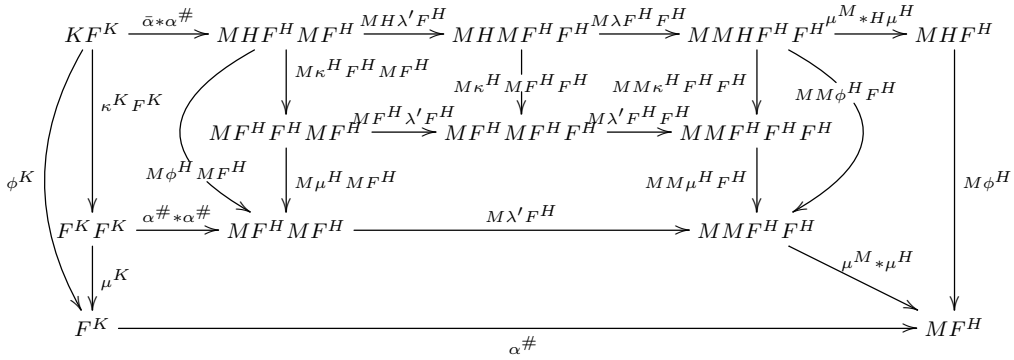
The following Lemma is a variant of Lemma 4.7 from [MM06].

Lemma 8.49 (cf. [MM06], Lemma 4.7)**.** *Let K be a set functor and let $\alpha : K \to MF^H$ be a natural transformation which factors as $\alpha = M\phi^H \cdot \bar{\alpha}$ for some $\bar{\alpha} : K \to MHF^H$. Then the following diagram commutes for the*

unique monad morphism $\alpha^\# : F^K \to MF^H$ *with* $\alpha^\# \cdot \kappa^K = \alpha$*:*



Proof. The diagram



commutes since all inner parts commute: the three curved arrows are just decomposed according to Lemma 2.24; the upper left-hand part commutes by $\alpha^\# \cdot \kappa^K = \alpha = M\phi^H \cdot \bar{\alpha}$, the lower part since $\alpha^\#$ is a monad morphism and the right-hand part since $\phi^H \cdot H\mu^H = \mu^H \cdot \phi^H F^H$ (see Remark 2.23). For the remaining three small parts use naturality of κ^H, Lemma 8.4 and one of the axioms for λ'. □

Theorem 8.50. *Under Assumption 7.16, every guarded M-RPS has a unique uninterpreted solution.*

Proof. The proof uses similar ideas as the one of Theorem 6.5 from [MM06]. The given guarded M-RPS $e : V \to MF$ factors through a natural transformation $e' : V \to MHF$, thus we obtain a natural transformation $h : F \to MF^H$ as in Definition 8.45. We define

$$e^\dagger = (\ V \xrightarrow{\mathsf{inr}} H + V \xrightarrow{\kappa^{H+V}} F \xrightarrow{h} MF^H\ ) \tag{8.28}$$

and prove that this is the unique solution of e.

(1) $e^\dagger$ solves e. First we compute

$$\begin{aligned}
p \cdot \kappa^{H+V} &= \mathsf{can} \cdot (MH\mu^{H+V} + \mathrm{id}) \cdot ([\eta^M HF \cdot H\eta^{H+V}, e']F + \eta^M) \\
&\quad \cdot [\phi^{H+V}, \eta^{H+V}]^{-1} \cdot \kappa^{H+V} \\
&\qquad\qquad \text{(by the definition of } p \text{ from Definition 8.45)} \\
&= \mathsf{can} \cdot (MH\mu^{H+V} + \mathrm{id}) \cdot ([\eta^M HF \cdot H\eta^{H+V}, e']F + \eta^M) \cdot \mathsf{inl} \\
&\quad \cdot (H+V)\eta^{H+V} \\
&\qquad\qquad \text{(by the definition of } \kappa^{H+V}\text{, see Theorem 2.22)} \\
&= M\mathsf{inl} \cdot MH\mu^{H+V} \cdot [\eta^M HF \cdot H\eta^{H+V}, e']F \cdot (H+V)\eta^{H+V} \\
&= M\mathsf{inl} \cdot [\eta^M HF \cdot H\eta^{H+V}, e'] \qquad \text{(by naturality of} \\
&\qquad [\eta^M HF \cdot H\eta^{H+V}, e'] \text{ and one of the unit laws for } F)
\end{aligned}$$

and

$$\begin{aligned}
h \cdot \kappa^{H+V} &= M[\phi^H, \eta^H] \cdot \mu^M(HF^H + \mathrm{Id}) \cdot M\mathsf{can} \cdot M(\lambda F^H + \eta^M) \\
&\quad \cdot M(Hh + \mathrm{id}) \cdot p \cdot \kappa^{H+V} \\
&\qquad\qquad \text{(by the definition of } h \text{ from Definition 8.45)} \\
&= M[\phi^H, \eta^H] \cdot \mu^M(HF^H + \mathrm{Id}) \cdot M\mathsf{can} \cdot M(\lambda F^H + \eta^M) \\
&\quad \cdot M(Hh + \mathrm{id}) \cdot M\mathsf{inl} \cdot [\eta^M HF \cdot H\eta^{H+V}, e'] \\
&\qquad\qquad\qquad\qquad \text{(by the above equation)} \\
&= M[\phi^H, \eta^H] \cdot \mu^M(HF^H + \mathrm{Id}) \cdot MM\mathsf{inl} \cdot M\lambda F^H \cdot MHh \\
&\quad \cdot [\eta^M HF \cdot H\eta^{H+V}, e'] \\
&= M\phi^H \cdot \mu^M HF^H \cdot M\lambda F^H \cdot MHh \cdot [\eta^M HF \cdot H\eta^{H+V}, e'] \\
&= M\phi^H \cdot [\mu^M HF^H \cdot M\lambda F^H \cdot MHh \cdot \eta^M HF \cdot H\eta^{H+V}, \qquad (8.29) \\
&\quad \mu^M HF^H \cdot M\lambda F^H \cdot MHh \cdot e'] \\
&= M\phi^H \cdot [\lambda F^H \cdot Hh \cdot H\eta^{H+V}, \mu^M HF^H \cdot M\lambda F^H \cdot MHh \\
&\quad \cdot e'] \\
&\qquad \text{(by naturality of } \eta^M \text{ and one of the unit laws for } M) \\
&= M\phi^H \cdot [\lambda F^H \cdot H\eta^M F^H \cdot H\eta^H, \mu^M HF^H \cdot M\lambda F^H \cdot MHh \\
&\quad \cdot e'] \qquad\qquad \text{(by the first monad morphism} \\
&\qquad\qquad\qquad\qquad \text{law for } h\text{, see Proposition 8.46)} \\
&= M\phi^H \cdot [\eta^M HF^H \cdot H\eta^H, \mu^M HF^H \cdot M\lambda F^H \cdot MHh \cdot e'] \\
&\qquad\qquad\qquad\qquad \text{(by the law for } \lambda \text{ and } \eta^M).
\end{aligned}$$

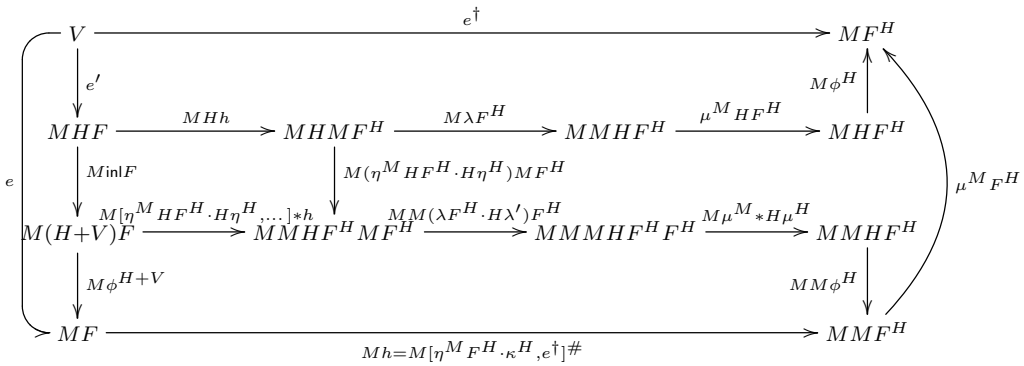
From (8.29) we see that it holds

$$\begin{aligned} h \cdot \kappa^{H+V} \cdot \mathsf{inl} &= M\phi^H \cdot [\eta^M HF^H \cdot H\eta^H, \mu^M HF^H \cdot M\lambda F^H \cdot MHh \cdot e'] \cdot \mathsf{inl} \\ &= M\phi^H \cdot \eta^M HF^H \cdot H\eta^H \\ &= \eta^M F^H \cdot \kappa^H \qquad \text{(by naturality of } \eta^M \text{ and the} \\ &\qquad\qquad\qquad\qquad\qquad\qquad \text{definition of } \kappa^H\text{, see Theorem 2.22).} \end{aligned}$$

Together with (8.28) it follows $h \cdot \kappa^{H+V} = [\eta^M F^H \cdot \kappa^H, e^\dagger]$, thus h is the unique monad morphism $[\eta^M F^H \cdot \kappa^H, e^\dagger]^\#$ with that property. From (8.28) and (8.29) we also get

$$\begin{aligned} e^\dagger &= h \cdot \kappa^{H+V} \cdot \mathsf{inr} \\ &= M\phi^H \cdot [\eta^M HF^H \cdot H\eta^H, \mu^M HF^H \cdot M\lambda F^H \cdot MHh \cdot e'] \cdot \mathsf{inr} \\ &= M\phi^H \cdot \mu^M HF^H \cdot M\lambda F^H \cdot MHh \cdot e' \end{aligned}$$

which makes the upper part of the diagram



commute. The left-hand part is the guardedness of e, the square in the left center is trivial, and for the lower part (the dots in the right copairing component stand for $\mu^M HF^H \cdot M\lambda F^H \cdot MHh \cdot e'$) remove M and use Lemma 8.49 where $K = H + V$, $\alpha = [\eta^M F^H \cdot \kappa^H, e^\dagger]$ and $\bar{\alpha} = [\eta^M HF^H, \mu^M HF^H] \cdot (H\eta^H + M\lambda F^H \cdot MHh \cdot e')$. Finally, commutativity

of the right-hand part is not difficult to see:

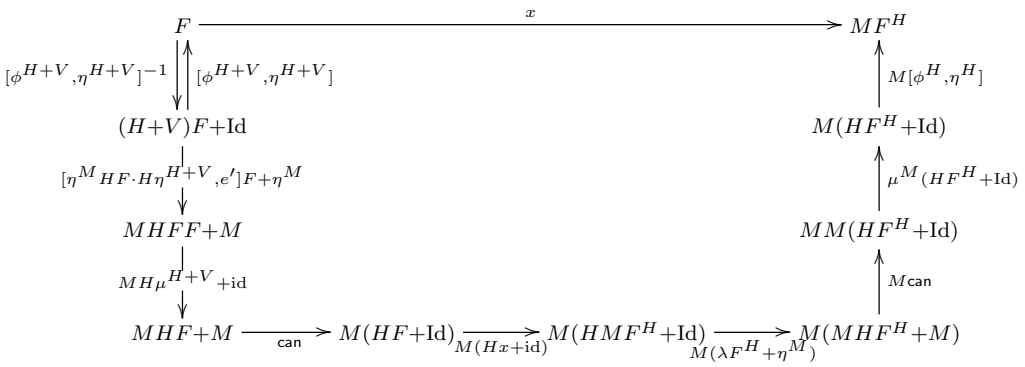
$$\begin{aligned}
&\mu^M F^H \cdot MM\phi^H \cdot M\mu^M HF^H \cdot MMMH\mu^H \cdot MM\lambda F^H F^H \\
&\cdot MMH\lambda' F^H \cdot M\eta^M HF^H MF^H \cdot MH\eta^H MF^H \\
&= \mu^M F^H \cdot MM\phi^H \cdot MMH\mu^H \cdot M\lambda F^H F^H \cdot MH\lambda' F^H \cdot MH\eta^H MF^H \\
&\qquad\qquad \text{(by naturality of } \eta^M \text{ and one of the unit laws for } M) \\
&= \mu^M F^H \cdot MM\phi^H \cdot MMH\mu^H \cdot M\lambda F^H F^H \cdot MHM\eta^H F^H \\
&\qquad\qquad\qquad\qquad \text{(by the axiom for } \lambda' \text{ and } \eta^H) \\
&= \mu^M F^H \cdot MM\phi^H \cdot M\lambda F^H \\
&\qquad\qquad \text{(by naturality of } \lambda \text{ and one of the unit laws for } F^H) \\
&= M\phi^H \cdot \mu^M HF^H \cdot M\lambda F^H \qquad \text{(by Remark 2.23)}
\end{aligned}$$

Thus the outside of the diagram commutes proving $e^\dagger$ to be a solution of e.

(2) $e^\dagger$ is the unique solution. Suppose $s : V \to MF^H$ is any solution of e and form $x = [\eta^M F^H \cdot \kappa^H, s]^\# : F \to MF^H$. It suffices to show that x is a coalgebra homomorphism between p and the final $\bar{\mathcal{H}}$-coalgebra from Theorem 8.30. Since h is known to be the unique such homomorphism, we have $h = x$ and conclude

$$\begin{aligned}
e^\dagger &= \mu^M F^H \cdot M[\eta^M F^H \cdot \kappa^H, e^\dagger]^\# \cdot e && \text{(since } e^\dagger \text{ is a solution of } e) \\
&= \mu^M F^H \cdot Mh \cdot e && \text{(from part (1) of the proof)} \\
&= \mu^M F^H \cdot Mx \cdot e && \text{(since } h = x) \\
&= s && \text{(since } s \text{ is a solution of } e)
\end{aligned}$$

using the definition of an uninterpreted solution (see Definition 8.48) twice. Clearly x is a morphism from $[\mathsf{Set}, \mathsf{Set}]_{\mathcal{M}}$. We only have to show that diagram (8.26) commutes with h replaced by x. In $[\mathsf{Set}, \mathsf{Set}]$ this diagram is



according to the definitions of h and p in Definition 8.45. We reverse the isomorphism $[\phi^{H+V}, \eta^{H+V}]^{-1}$ as shown in the diagram and consider both coproduct components of $(H+V)F+\mathrm{Id}$ separately: the right-hand component

commutes due to the first law $x \cdot \eta^{H+V} = \eta^M F^H \cdot \eta^H$ for the monad morphism x, one of the unit laws for M and naturality of η^M. For the left-hand component we start our calculation with the longer lower path in the diagram and use the second law for the monad morphism x:

$$
\begin{aligned}
& M\phi^H \cdot \mu^M HF^H \cdot M\lambda F^H \cdot MHx \cdot MH\mu^{H+V} \cdot [\eta^M HF \cdot H\eta^{H+V}, e']F \\
&= M\phi^H \cdot \mu^M HF^H \cdot M\lambda F^H \cdot MH(\mu^M * \mu^H) \cdot MHM\lambda' F^H \cdot MH(x * x) \\
&\quad \cdot [\eta^M HF \cdot H\eta^{H+V}, e']F \qquad \text{(by the second monad morphism law for } x) \\
&= M\phi^H \cdot \mu^M HF^H \cdot M\mu^M HF^H \cdot MM\lambda F^H \cdot M\lambda MF^H \cdot MHMM\mu^H \\
&\quad \cdot MHM\lambda' F^H \cdot MH(x * x) \cdot [\eta^M HF \cdot H\eta^{H+V}, e']F \\
&\qquad\qquad \text{(by the axiom for } \lambda \text{ and } \mu^M) \\
&= \mu^M F^H \cdot M\mu^M F^H \cdot MMM\phi^H \cdot MM\lambda F^H \cdot MMHM\mu^H \cdot MMH\lambda' F^H \\
&\quad \cdot MMHF^H x \cdot M\lambda F^H F \cdot MHxF \cdot [\eta^M HF \cdot H\eta^{H+V}, e']F \\
&\qquad\qquad \text{(by naturality of } \mu^M \text{ and } \lambda) \\
&= (\mu^M * \mu^H) \cdot M\mu^M F^H F^H \cdot MMM\phi^H F^H \cdot MM\lambda F^H F^H \cdot MMH\lambda' F^H \\
&\quad \cdot MMHF^H x \cdot M\lambda F^H F \cdot MHxF \cdot [\eta^M HF \cdot H\eta^{H+V}, e']F \\
&\qquad\qquad \text{(by naturality of } \lambda\text{, Remark 2.23 and naturality of } \mu^M) \\
&= (\mu^M * \mu^H) \cdot M\mu^M F^H F^H \cdot MM\lambda' F^H \cdot MM\phi^H MF^H \cdot MMHF^H x \\
&\quad \cdot M\lambda F^H F \cdot MHxF \cdot [\eta^M HF \cdot H\eta^{H+V}, e']F \qquad \text{(by (8.1))} \\
&= (\mu^M * \mu^H) \cdot \mu^M MF^H F^H \cdot MM\lambda' F^H \cdot MMF^H x \cdot MM\phi^H F \cdot M\lambda F^H F \\
&\quad \cdot MHxF \cdot [\eta^M HF \cdot H\eta^{H+V}, e']F \\
&\qquad\qquad \text{(by the multiplication law for } M \text{ and naturality of } \phi^H) \\
&= (\mu^M * \mu^H) \cdot M\lambda' F^H \cdot MF^H x \cdot \mu^M F^H F \cdot MM\phi^H F \cdot M\lambda F^H F \cdot MHxF \\
&\quad \cdot [\eta^M HF \cdot H\eta^{H+V}, e']F \qquad \text{(by naturality of } \mu^M) \\
&= (\mu^M * \mu^H) \cdot M\lambda' F^H \cdot MF^H x \cdot [\mu^M F^H \cdot MM\phi^H \cdot M\lambda F^H \cdot MHx \\
&\quad \cdot \eta^M HF \cdot H\eta^{H+V}, \mu^M F^H \cdot MM\phi^H \cdot M\lambda F^H \cdot MHx \cdot e']F \\
&\overset{(*)}{=} (\mu^M * \mu^H) \cdot M\lambda' F^H \cdot MF^H x \cdot [M\phi^H \cdot \lambda F^H \cdot Hx \cdot H\eta^{H+V}, s]F \\
&\qquad\qquad \text{(see below)} \\
&= (\mu^M * \mu^H) \cdot M\lambda' F^H \cdot MF^H x \cdot [M\phi^H \cdot \lambda F^H \cdot H\eta^M F^H \cdot H\eta^H, s]F \\
&\qquad\qquad \text{(by the first monad morphism law for } x) \\
&= (\mu^M * \mu^H) \cdot M\lambda' F^H \cdot MF^H x \cdot [\eta^M F^H \cdot \phi^H \cdot H\eta^H, s]F \\
&\qquad\qquad \text{(by the axiom for } \lambda \text{ and } \eta^M \text{ and naturality of } \eta^M)
\end{aligned}
$$

$$
\begin{aligned}
&= (\mu^M * \mu^H) \cdot M\lambda' F^H \cdot MF^H x \cdot [\eta^M F^H \cdot \kappa^H, s]F \\
&\qquad\qquad \text{(by the definition of } \kappa^H\text{, see Theorem 2.22)} \\
&= (\mu^M * \mu^H) \cdot M\lambda' F^H \cdot MF^H x \cdot xF \cdot \kappa^{H+V} F \\
&\qquad\qquad \text{(by its definition } x \text{ extends } [\eta^M F^H \cdot \kappa^H, s]\text{)} \\
&= x \cdot \mu^{H+V} \cdot \kappa^{H+V} F \qquad \text{(by the second monad morphism law for } x\text{)} \\
&= x \cdot \phi^{H+V} \qquad\qquad \text{(by Lemma 2.24)}
\end{aligned}
$$

For the equality marked with $(*)$ we use the equation $s = \mu^M F^H \cdot MM\phi^H \cdot M\lambda F^H \cdot MHx \cdot e'$ which follows from the fact that s is a solution of the guarded M-RPS e as we see in

$$
\begin{aligned}
s &= \mu^M F^H \cdot Mx \cdot e \qquad\qquad \text{(since } s \text{ is a solution of } e\text{)} \\
&= \mu^M F^H \cdot Mx \cdot M\phi^{H+V} \cdot M\mathsf{inl}F \cdot e' \qquad \text{(by guardedness of } e\text{)} \\
&= \mu^M F^H \cdot Mx \cdot M\mu^{H+V} \cdot M\kappa^{H+V} F \cdot M\mathsf{inl}F \cdot e' \qquad \text{(by Lemma 2.24)} \\
&= \mu^M F^H \cdot M(\mu^M * \mu^H) \cdot MM\lambda' F^H \cdot M(x * x) \cdot M\kappa^{H+V} F \cdot M\mathsf{inl}F \cdot e' \\
&\qquad\qquad \text{(by the second monad morphism law for } x\text{)} \\
&= \mu^M F^H \cdot M(\mu^M * \mu^H) \cdot MM\lambda' F^H \cdot M[\eta^M F^H \cdot \kappa^H, s]MF^H \cdot M(H+V)x \\
&\quad \cdot M\mathsf{inl}F \cdot e' \qquad \text{(by its definition } x \text{ extends} [\eta^M F^H \cdot \kappa^H, s]\text{)} \\
&= \mu^M F^H \cdot M(\mu^M * \mu^H) \cdot MM\lambda' F^H \cdot M\eta^M F^H MF^H \cdot M\kappa^H MF^H \cdot MHx \\
&\quad \cdot e' \\
&= \mu^M F^H \cdot MM\mu^H \cdot M\lambda' F^H \cdot M\kappa^H MF^H \cdot MHx \cdot e' \\
&\qquad\qquad \text{(by naturality of } \eta^M \text{ and one of the unit laws for } M\text{)} \\
&= \mu^M F^H \cdot MM\phi^H \cdot M\lambda F^H \cdot MHx \cdot e' \qquad \text{(by Lemmata 8.4 and 2.24).}
\end{aligned}
$$

Finally, the whole diagram commutes showing that x is an $\bar{\mathcal{H}}$-coalgebra homomorphism between p and the final coalgebra from Theorem 8.30 as desired. □

Since Assumption 7.16 holds true for any combination of a finitary polynomial functor H and one of the monads $M = \mathrm{Id} + 1$, $M = \mathcal{P}$ or $M = \mathcal{D}$ as explained in Example 7.18(1), we obtain the following

Corollary 8.51. *Every guarded* $(\mathrm{Id} + 1)$*-RPS (partial RPS),* $\mathcal{P}$*-RPS (nondeterministic RPS) and* $\mathcal{D}$*-RPS (probabilistic RPS) where* H *is finitary polynomial has a unique uninterpreted solution.*

Notice that by the use of the free monad F^H in Definition 8.48, for finitary polynomial functors H this means that uninterpreted solutions are unique w. r. t. *finite* trees—there may be further solutions that contain infinite trees. In the remainder of Section 8.5 we shall use the free CIM T^H instead of F^H which means for finitary polynomial functors H that we consider uninterpreted solutions w. r. t. *possibly infinite* trees.

Remark 8.52. For the definitions and results in Section 8.5.1 the underlying distributive law λ does not necessarily need to be the canonical one, cf. Remark 8.47. It would be sufficient to require in Definition 8.48 an arbitrary distributive law of H over the monad M and that H and $H + V$ have free algebras. However, widening Definition 8.48 in this respect, one would have to consider "λ-RPS's" instead of "M-RPS's" since there might be several distributive laws of a functor H over a monad M. For simplicity of presentation and since we have seen (Example 6.14,(1)–(3)) that the canonical distributive laws are the desired "effect handlers", we restricted ourselves to those. Moreover, as remarked below Example 6.23, distributive laws other than the canonical ones seem to be rare.

8.5.2 Nonempty Nondeterministic, Composite and Plain RPS's

Definition 8.53. Let H and V be endofunctors on Set. A *nonempty nondeterministic recursive program scheme* (*composite recursive program scheme*) is a natural transformation $e : V \to MF$ where $M = \mathcal{P}^+$ ($M = (-)^E$) and where H and V must be finitary polynomial functors (H must be iteratable and $H + V$ must have free algebras). It is called *guarded* if it factors as follows:

$$e \equiv (V \xrightarrow{e'} MHF \xrightarrow{M\mathsf{in}\mathsf{l}F} M(H+V)F \xrightarrow{M\phi^{H+V}} MF)\,.$$

An *uninterpreted solution* of e is a natural transformation $e^\dagger : V \to MT^H$ such that the diagram

$$\begin{array}{ccc} V & \xrightarrow{\quad e^\dagger \quad} & MT^H \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle \mu^M T^H} \\ MF & \xrightarrow[M[\eta^M T^H \cdot \kappa^H, e^\dagger]^\#]{} & MMT^H \end{array} \tag{8.30}$$

commutes. Here $[\eta^M T^H \cdot \kappa^H, e^\dagger]^\#$ is the unique monad morphism into MT^H extending $[\eta^M T^H \cdot \kappa^H, e^\dagger]$ (see Definition 2.8) where the composite monad MT^H is induced by the extension (see Proposition 8.12 for $\mathcal{P}^+$ and Corollary 8.7 for $(-)^E$) of the canonical distributive law (see Example 6.23,(1)–(2)).

Remarks 8.54. 1. Definition 8.53 mainly is a Kleisli category variant of Definition 2.52 of "ordinary RPS's". However, notice that T^{H+V} and τ^{H+V} are replaced by F and ϕ^{H+V} here.

2. For polynomial functors H and V and any set X, FX is the set of all finite trees or terms built from the operation symbols from the signatures corresponding to H and V and the variables from X (see Example 2.9). Explicitly, $[\eta^+ T^H \cdot \kappa^H, e^\dagger]^\#_X$ performs a nondeterministic (compositional) variant of second-order substitution in trees (cf. [MM06], Section 4.1).

Remark 8.55. In case $E = 1$ (i. e. $M = \mathrm{Id}$), we speak about *plain recursive program schemes*. We compare the plain RPS's from Definition 8.53 with the existing notion of RPS's from Definition 2.52:

1. Plain RPS's are a special case of the RPS's from Definition 2.52 where we restrict to $\mathcal{C} = \mathsf{Set}$ and to finite terms on the right-hand sides of RPS's. More precisely, a plain RPS is a natural transformation $e : V \to F$ which can be viewed as a RPS $\mathsf{in} \cdot e : V \to T^{H+V}$ where $\mathsf{in} = (\kappa^{H+V})^\# : F \to T^{H+V}$. If e is guarded in the sense of Definition 8.53 (using $M = \mathrm{Id}$), then $\mathsf{in} \cdot e$ is guarded in the sense of Definition 2.52.

2. Since the "composite monad" $\mathrm{Id} \cdot T^H$ is the CIM T^H (see Observation 8.28), the notion of an uninterpreted solution also becomes a special case of the one from Definition 2.52 (as far as ideal natural transformations $e^\dagger$ are concerned). In fact, it holds $[\kappa^H, e^\dagger]^\# \cdot \mathsf{in} = [\kappa^H, e^\dagger]^\# : F \to T^H$ by the uniqueness of such monad morphisms extending $\kappa^{H+V} : H + V \to F$.

3. The assumption on uninterpreted solutions of RPS's to be ideal is necessary to ensure the existence of $[\kappa^H, e^\dagger]^\# : T^{H+V} \to T^H$. Working with finite terms in Definition 8.53 has the advantage that we can drop this assumption. In case of a guarded plain RPS $e : V \to F$ uninterpreted solutions automatically are ideal.

4. Technically, the restriction to finite terms on the right-hand sides of nonempty nondeterministic, composite or plain RPS's is due to the fact that the monad $\mathcal{P}^+ T^H$ is not a CIM and we thus cannot exploit freeness of the CIM T^{H+V}. However, since by Theorem 8.23 (and Lemma 8.26) $\mathcal{P}^+ T^H$ is an idealized monad together with a solution operation $(-)^\dagger$ giving canonical (greatest) solutions for guarded equation morphisms, it comes close to a CIM. In order to capture infinite terms, it would be interesting to see whether $\mathcal{P}^+ T^H$ is something like a "complete Elgot monad" and whether the free CIM T^{H+V} also is the "free complete

Elgot monad". But whereas the concept of Elgot monads has recently been investigated [AMV11], there exist no results for complete Elgot monads.

Nonempty Nondeterministic RPS's

We shall first treat nonempty nondeterministic RPS's (or *NNRPS*, for short). Unlike nondeterministic RPS's (or $\mathcal{P}$-RPS's, see Definition 8.48) NNRPS do not allow empty sets on the right-hand sides of equations or in solutions. It follows that they form a subset of $\mathcal{P}$-RPS's and that certain NNRPS's (like for example the one given by $v(x) = \{h(v(x))\}$) lack an uninterpreted solution in finite trees. Indeed, this is always the case if the unique uninterpreted solution according to Corollary 8.51 (viewing the NNRPS as a $\mathcal{P}$-RPS) involves the empty set. As we shall see in Theorem 8.60 below, this lack is remedied by using the free CIM T^H in Definition 8.53, i. e. by taking solutions in possibly infinite trees. We shall also see in Remark 8.57 that NNRPS's come closer to classical nondeterministic recursive program schemes than $\mathcal{P}$-RPS's.

Example 8.56. Consider the NNRPS (II.1) from the introduction to Part II. It is formulated in the classical way using the special binary function symbol or, see e. g. [AN80], Section II. This can be viewed as a natural transformation $e : V \to \mathcal{P}^+T^{H+V}$ as follows: according to the signatures of new and given function symbols, we choose the finitary polynomial set functors $VX = X$ and $HX = X$. We translate the symbol or into a two-element set containing the two arguments of the symbol and abstract away from a concrete variable set, obtaining the natural transformation e given by $e_X(\phi(x)) = \{x, \mathsf{op}(\phi(x))\}$ for every set X. The naturality states that it is invariant under renaming of the variable x.
In classical terms, the NNRPS (II.1) is a *Greibach scheme* since every new function symbol that occurs on the right-hand side of the equation is part of a term headed by a given function symbol, see e. g. [AN80], Section IV. Correspondingly, the natural transformation e is (nearly) guarded since every element (except the variable x) of the right-hand set is a term headed by a given operation symbol. In Remark 8.57 we explain how a guarded natural transformation can be achieved.
Let us denote the infinite set (II.2) from the introduction to Part II by S_X. We obtain the natural transformation $e^\dagger$ given by $e^\dagger_X(\phi(x)) = S_X$ for every set X. Using Remark 8.54(2), we see that diagram (8.30) commutes; thus $e^\dagger$ is an uninterpreted solution of e. Similarly, the natural transformation s given by $s_X(\phi(x)) = S_X \setminus \{t\}$ for every set X is an uninterpreted solution of e, where t is the only infinite tree from S_X (the rightmost one in (II.2)).

Remark 8.57. More generally, every classical NNRPS in the sense of Arnold and Nivat [AN80] can be translated into a NNRPS in the sense of Definition 8.53, using the following ideas:

- the set functors V and H are the finitary polynomial functors expressing the signatures of new and given function symbols;
- every occurrence of the function symbol or is translated to a two-element set;
- the given function symbols are distributed over sets using the canonical distributive law $\lambda : H\mathcal{P}^+ \to \mathcal{P}^+H$;
- nested sets are flattened using $\mu^+ : \mathcal{P}^+\mathcal{P}^+ \to \mathcal{P}^+$;
- for every set S occurring in a term headed by a new function symbol, an additional new function symbol $\phi_S(x_1, \ldots, x_n)$ of arity according to the number n of variables in S is introduced, S is replaced by $\phi_S(x_1, \ldots, x_n)$ and the equation $\phi_S(x_1, \ldots, x_n) = S$ is added to the NNRPS;
- occurrences of single variables x_i in sets are replaced by $\pi_i(x_1, \ldots, x_n)$ where π_i is an additional given function symbol and the $x_1, \ldots, x_n$ are all variables occurring in the elements of the set (the idea is of course that π_i denotes the i-th projection);
- the natural transformation $e : V \to \mathcal{P}^+F$ constituting the NNRPS is given for every set X and every element from VX by the right-hand side of the equation for the corresponding new function symbol.

In order to obtain a guarded NNRPS from a classical Greibach scheme, it might be necessary to substitute some new function symbols by the right-hand sides of their equations. In conclusion, our notion of a NNRPS covers the classical one, and classical Greibach schemes translate to guarded NNRPS's. Moreover, our notion generalizes the classical one: whereas the classical NNRPS's from [AN80] define finitely many new operations, and, more important, only allow for finite (nonempty) sets of finite terms on the right-hand sides, our approach also captures infinitely many newly defined operations and arbitrary (nonempty) sets. It might even be possible to generalize our approach to infinite terms on the right-hand sides, see Remark 8.55(4).

Example 8.58. We illustrate the translation of a classical NNRPS to a NNRPS in our sense. Consider the following classical NNRPS:

$$\begin{aligned}\phi(x,y) &= f(x,y) \text{ or } g(\phi(y,x) \text{ or } f(y,x))\\ \psi(x,y) &= g(\phi(x \text{ or } \psi(g(x) \text{ or } y, f(x,x)), f(y,x)))\end{aligned}$$

First, every occurrence of or is substituted by the two-element set containing the operands:

$$\begin{aligned}\phi(x,y) &= \{f(x,y), g(\{\phi(y,x), f(y,x)\})\}\\ \psi(x,y) &= g(\phi(\{x, \psi(\{g(x), y\}, f(x,x))\}, f(y,x)))\end{aligned}$$

Since in the second equation sets appear twice in terms headed by new function symbols, we introduce two new function symbols:

$$\begin{aligned}\phi(x,y) &= \{f(x,y), g(\{\phi(y,x), f(y,x)\})\}\\ \psi(x,y) &= g(\phi(\chi(x,y), f(y,x)))\\ \chi(x,y) &= \{x, \psi(\rho(x,y), f(x,x))\}\\ \rho(x,y) &= \{g(x), y\}\end{aligned}$$

Next we make all right-hand sides sets of terms that contain no sets. For the first equation, we use λ to distribute the given function symbol g over the inner set and then flatten the nested set by making $f(x,y)$ a singleton set and using μ^+. For the second equation, we make the right-hand term a singleton set. And the third and fourth equation already are in the desired form.

$$\begin{aligned}\phi(x,y) &= \{f(x,y), g(\phi(y,x)), g(f(y,x))\}\\ \psi(x,y) &= \{g(\phi(\chi(x,y), f(y,x)))\}\\ \chi(x,y) &= \{x, \psi(\rho(x,y), f(x,x))\}\\ \rho(x,y) &= \{g(x), y\}\end{aligned}$$

The original NNRPS is a Greibach scheme; in order to obtain a guarded NNRPS, we substitute the elements x and y in the third and fourth equation by $\pi_1(x,y)$ and $\pi_2(x,y)$, and use the second equation to rewrite the element $\psi(\rho(x,y), f(x,x))$ starting with a new function symbol in the third equation (and then flatten the nested set on the right-hand side of the third equation).

$$\begin{aligned}\phi(x,y) &= \{f(x,y), g(\phi(y,x)), g(f(y,x))\}\\ \psi(x,y) &= \{g(\phi(\chi(x,y), f(y,x)))\}\\ \chi(x,y) &= \{\pi_1(x,y), g(\phi(\chi(\rho(x,y), f(x,x)), f(f(x,x), \rho(x,y))))\}\\ \rho(x,y) &= \{g(x), \pi_2(x,y)\}\end{aligned}$$

According the signature of the new function symbols ϕ, ψ, χ and ρ (all binary) we choose the finitary polynomial set functor $VX = (X \times X)^4$; according to the signature of the given function symbols f, g, π_1 and π_2 (g unary, all others binary) we choose the finitary polynomial set functor $HX = X \times X + X + X \times X + X \times X$. Now we can read off the natural transformation $e : V \to \mathcal{P}^+F$ from the above equations for each coproduct component of V separately, e. g. for the first component corresponding to ϕ, e is given by $e_X(\phi(x, y)) = \{f(x, y), g(\phi(y, x)), g(f(y, x))\}$ for every set X.

Lemma 8.59 (cf. [MM06], Lemma 4.7)**.** *Let K be a set functor and let $\alpha : K \to \mathcal{P}^+T^H$ be an ideal natural transformation, i. e. $\alpha = \mathcal{P}^+\tau^H \cdot \bar{\alpha}$ for some $\bar{\alpha} : K \to \mathcal{P}^+HT^H$. Then the following diagram commutes for the unique monad morphism $\alpha^\# : F^K \to \mathcal{P}^+T^H$ with $\alpha^\# \cdot \kappa^K = \alpha$:*

$$\begin{array}{ccccccccc}
KF^K & \xrightarrow{\bar{\alpha} * \alpha^\#} & \mathcal{P}^+HT^H\mathcal{P}^+T^H & \xrightarrow{\mathcal{P}^+H\lambda' T^H} & \mathcal{P}^+H\mathcal{P}^+T^HT^H & \xrightarrow{\mathcal{P}^+\lambda T^HT^H} & \mathcal{P}^+\mathcal{P}^+HT^HT^H & \xrightarrow{\mu^+ * H\mu^H} & \mathcal{P}^+HT^H \\
{\scriptstyle \phi^K}\downarrow & & & & & & & & \downarrow{\scriptstyle \mathcal{P}^+\tau^H} \\
F^K & & & & & & & & \mathcal{P}^+T^H \\
& & & & & {\scriptstyle \alpha^\#} & & & \\
\end{array}$$

Proof. The proof is the same as for Lemma 8.49 except that M is replaced by $\mathcal{P}^+$, F^H and ϕ^H are replaced by T^H and τ^H and instead of Lemma 2.24 and Remark 2.23, Lemma 2.49 and Remark 2.48 are invoked. Instead of Lemma 8.4 use Remark 8.13. □

Theorem 8.60. *Every guarded NNRPS has a canonical greatest uninterpreted solution.*

Proof. The proof is a variant of the above proof of Theorem 8.50 where instead of a guarded M-RPS, a guarded NNRPS $e : V \to \mathcal{P}^+F$ is considered. In the proof M is replaced by $\mathcal{P}^+$, F^H and ϕ^H are replaced by T^H and τ^H. Instead of Definition 8.45 including (8.26), Proposition 8.46 and Lemma 8.49, we use Definition 8.37 including (8.10), Proposition 8.40 and Lemma 8.59. Lemma 8.4 is replaced by Remark 8.13, Theorem 2.22 by Theorem 2.47, Remark 2.23 by Remark 2.48, Definition 8.48 by Definition 8.53, (8.1) by (8.2) with $M = \mathcal{P}^+$ and Lemma 2.24 by Lemma 2.49.

Part (2) of the proof no longer shows uniqueness of solutions, but that the solution $e^\dagger$ defined in the existence part (1) of the proof is a (componentwise) greatest solution. Indeed, weak finality of $J[\tau^H, \eta^H]^{-1}$ (Theorem 8.33) is invoked instead of finality of $J[\phi^H, \eta^H]^{-1}$ (Theorem 8.30), thus for any solution $s : V \to \mathcal{P}^+T^H$ of e it suffices to show that $x = [\eta^+T^H \cdot \kappa^H, s]^\# : F \to \mathcal{P}^+T^H$ is a coalgebra homomorphism between p and the weakly final $\bar{\mathcal{H}}$-coalgebra:

since h is known to be the componentwise greatest such homomorphism, it follows $h_X \geq x_X$ for every set X and we conclude

$$e^\dagger_X = \mu^+_{T^H X} \cdot \mathcal{P}^+[\eta^+ T^H \cdot \kappa^H, e^\dagger]^\#_X \cdot e_X = \mu^+_{T^H X} \cdot \mathcal{P}^+ h_X \cdot e_X \geq \mu^+_{T^H X} \cdot \mathcal{P}^+ x_X \cdot e_X = s_X$$

for every set X using Definition 8.53 (for $M = \mathcal{P}^+$) and monotonicity of composition in $\mathsf{Set}_{\mathcal{P}^+}$.

Finally, we know from part (1) of the proof of Theorem 8.33 that the components of h are given canonically by (8.4) where the e in (8.4) is taken to be p_X from (8.9), for the construction of which the factor e' of the given guarded NNRPS is used. It follows from (8.28) that also the solution $e^\dagger$ of the NNRPS is given canonically. □

Corollary 8.61. *For every uninterpreted solution $s : V \to \mathcal{P}^+ T^H$ of a NNRPS the sets of all finite cuttings of trees from $s_X(z)$ and $e^\dagger_X(z)$ are the same for every set X and every $z \in VX$.*

Proof. In the second part of the proof of Theorem 8.60 we prove that for every solution s of every guarded NNRPS e the monad morphism $[\eta^+ T^H \cdot \kappa^H, s]^\#$ is an $\bar{\mathcal{H}}$-coalgebra homomorphism. According to Lemma 8.34, we have the desired property from the statement for this $\bar{\mathcal{H}}$-coalgebra homomorphism and the $\bar{\mathcal{H}}$-coalgebra homomorphism h; this implies that this property also holds for their respective restrictions $s = [\eta^+ T^H \cdot \kappa^H, s]^\# \cdot \kappa^{H+V} \cdot \mathsf{inr}$ and $e^\dagger = h \cdot \kappa^{H+V} \cdot \mathsf{inr}$. □

Remark 8.62. The main result of Arnold and Nivat [AN80] is that greatest solutions of Greibach schemes give the "right" semantics of NNRPS's and can be computed as greatest fixed points. We confirmed the former in Theorem 8.60 and generalized it to a wider class of NNRPS's (cf. Remark 8.57). From our results we also easily recover the latter: restricting to finite sets on the right-hand sides of NNRPS's, the operator $h \mapsto \mathcal{P}^+[\tau^H, \eta^H] \cdot \mu^+(HT^H + \mathrm{Id}) \cdot \mathcal{P}^+ \mathsf{can} \cdot \mathcal{P}^+(\lambda T^H + \eta^+) \cdot \mathcal{P}^+(Hh + \mathrm{id}) \cdot p$ on $\mathsf{Set}(F, \mathcal{P}^+ T^H)$ given by equation (8.11) or equivalently by diagram (8.10) is componentwise continuous; since we know from Theorem 8.33 and Lemma 8.34 that the greatest fixed point of this operator exists, the second part of Arnold's and Nivat's result follows from (the dual of) Kleene's fixed point theorem. However, the operator is no longer continuous if we allow for infinite sets on the right-hand sides of NNRPS's.

Composite RPS's

Theorem 8.63. *Every guarded composite RPS has a unique uninterpreted solution.*

Proof. The proof is a variant of the above proof of Theorem 8.50 where instead of a guarded M-RPS, a guarded composite RPS $e : V \to F^E$ is considered. In the proof M is replaced by $(-)^E$, F^H and ϕ^H are replaced by T^H and τ^H. Instead of Definition 8.45 including (8.26), Proposition 8.46 and Lemma 8.49, we use Definition 8.41 including (8.25), Proposition 8.42 and Lemma 8.59 (with $\mathcal{P}^+$ replaced by $(-)^E$ and referring to Remark 8.9 instead of Remark 8.13). Lemma 8.4 is replaced by Remark 8.9, Theorem 2.22 by Theorem 2.47, Remark 2.23 by Remark 2.48, Definition 8.48 by Definition 8.53, (8.1) by (8.2) with $M = (-)^E$ and Lemma 2.24 by Lemma 2.49.

In the uniqueness part of the proof finality of $J[\tau^H, \eta^H]^{-1}$ (Theorem 8.35) is invoked instead of finality of $J[\phi^H, \eta^H]^{-1}$ (Theorem 8.30). □

For the special case $E = 1$ (i. e. $M = \mathrm{Id}$) of plain RPS's it follows

Corollary 8.64 (also follows from [MM06])**.** *Every guarded plain RPS has a unique uninterpreted solution.*

Remark 8.65. Recall from Remark 8.55 that plain RPS's are a subset of RPS's as defined in Definition 2.52. The latter possess unique uninterpreted solutions by Theorem 2.53. Thus Corollary 8.64 is no new result but shows that our framework recovers parts of the existing results on RPS's when applied to the identity monad $M = \mathrm{Id}$, i. e. when we consider plain RPS's.

8.6 Related Work

Different semantics of RPS's were investigated in the 1970's and 80's: for deterministic RPS's see for example Courcelle [Cou83], Guessarian [Gue81] and Nivat [Niv75]; for nondeterministic RPS's we mention Boudol [Bou80], Arnold and Nivat [AN80] and Poigné [Poi82]. We used [AN80] as a classical definition and result on NNRPS's to compare our definition and results with. A category theoretic approach to deterministic RPS's was given by Ghani, Lüth and De Marchi [GLM03] and by Milius and Moss [MM06]. We made use of several techniques from [MM06], especially in Section 8.5. However, in order to account for the effects, we could neither use the whole approach from loc. cit. nor were the existing techniques sufficient for our purposes. Goncharov and Schröder [GS13] present an approach parametric in a wide range of effects and obtain a unique solution result for their guarded corecursive schemes. However, besides a structure different to our schemes from Section 8.5, their schemes do not allow for arbitrary given operations.

Except for Lemma 8.15, which is a well-known result by Beck [Bec69], Chapter 8 contains results from our paper [Sch11], the short abstract [Sch10b]

and new material: nearly all parts dealing with the nonempty powerset monad $\mathcal{P}^+$ were published in [Sch11] without the full proofs. Also Definition 8.16, Lemma 8.19 (without full proof), Lemma 8.29 (with full proof), Remark 8.54(2) and Remark 8.55 can already be found in [Sch11]. Remark 8.14, Remarks 8.38, Example 8.39, Example 8.58 and Lemma 8.59 are inside parts dealing with $\mathcal{P}^+$ but were added. Proposition 8.2 (without proof) and a slight variant of Definition 8.48 were published in [Sch10b]. Also Lemma 8.29 and Theorem 8.30 were hinted in loc. cit.

Chapter 9

Conclusion

9.1 Summary

In this thesis, we presented many new category theoretic formats of recursive specifications. In order to define the corresponding notions of solutions and to prove that solutions uniquely exist, we combined algebras and coalgebras for a functor with distributive laws. We were able to demonstrate that several know results from different areas of computer science arise as special cases from our work. We dealt with systems of recursive equations as well as recursive program schemes, both of them in two different flavors according to which the thesis was split in two parts.

In the first part (Chapters 3 to 5) we used distributive laws of an algebra functor (possibly a pointed functor or a monad) over a coalgebra functor to define algebraic operations on final coalgebras; in particular we worked with abstract GSOS rules. This approach goes back to Turi and Plotkin [TP97], and Bartels [Bar04] has first results about unique solutions of a simple format of recursive specifications using such operations. We developed this further and presented here in a systematic way several formats of recursive systems of equations and of recursive program schemes using such operations. Most prominently, we introduced sandwiched recursive program schemes w. r. t. an abstract GSOS rule ℓ (or sandwiched ℓ-RPS's, for short). We showed how our formats capture and extend recursive specification formats from the literature such as Milner's CCS and Rutten's behavioral differential equations for streams. Our approach via CIAs enabled us to state our main result: we proved the highly desirable property of compositionality of the specification formats, thereby giving a formal explanation why such recursive specifications can be solved in a step-by-step manner.

In the second part (Chapters 6 to 8) we used distributive laws of algebra

functors over monads exhibiting different computational effects in the sense of Moggi [Mog91]. This way we were able to deal with five concrete computational effects in recursive specifications. This use of distributive laws seems to be new. We considered two notions of algebras with effects in which systems of recursive equations have unique solutions and characterized these algebras for three concrete effects. For all five effects, we worked out unique or canonical solution results for recursive program schemes with these effects. We put emphasis on the most interesting case of the "nonempty nondeterminism" effect. We saw how our results extend classical work on such recursive program schemes by Arnold and Nivat [AN80] and how they relate to the work of Milius and Moss [MM06] who investigated deterministic recursive program schemes category theoretically.

9.2 Future Work

Future Work: Part I

Part I of this thesis seems to us quite comprehensive when it comes to formats of recursive specifications on final coalgebras. However, there remains the question whether our results can be generalized to other algebras than the initial CIAs (alias final coalgebras) along the lines of Capretta, Uustalu and Vene [CUV06] who generalized Bartels' results from [Bar03, Bar04]. A different kind of generalization to be investigated concerns the step from using free monads in the definition of ℓ-(S)RPS's to arbitrary monads. And third, it would be of interest to extend the results of [MM09] on properties of recursive program scheme solutions to the richer settings of Chapter 5.

Besides the investigation of those general questions, our work in Part I obviously can be applied to arbitrary further final coalgebras than the five ones from Sections 4.2 and 5.2. For example, it is clear that our results can be used to obtain unique solutions of recursive specifications on weighted transition systems [Kli09]. It should also be interesting to investigate whether our results yield unique solution theorems for name and value passing process calculi as considered by Fiore and Turi [FT01].

As another task towards the practical application of our results it should be interesting to identify for the various final coalgebras concrete syntactic formats of operational rules that correspond to ℓ-(S)RPS's. For example, in the case of CCS processes, Turi and Plotkin [TP97] proved that abstract GSOS rules correspond precisely to transition system specifications with operational rules in the GSOS format of [BIM95]. Bartels gave in his thesis [Bar04] concrete syntactic rule formats for abstract GSOS rules in several

other concrete cases and Klin [Kli09] studied the instantiation of abstract GSOS rules for weighted transition systems. This existing body of work should be completed by spelling out concrete syntactic formats and proving that they are equivalently characterized by ℓ-(S)RPS's.

Finally, our excursion in Section 4.3 leaves some questions for future work: again it should be interesting to extend our work by considering further final coalgebras, e. g. infinite trees and non-well-founded sets. A starting point would be the notion of a rational infinite tree which can be found in Silva's and Rutten's paper [SR10] under the name "rational binary tree". And there are two concrete questions about classes of languages which can be defined by using different sets of operations in recursive specifications: (1) can non-regular languages be defined if the Kleene star operation is added to the regular language setting from Section 4.3.3 with the constant languages $\emptyset$, $\{\varepsilon\}$ and $\{a\}$ for every $a \in A$ as well as the operations of union, intersection, complement and prefixing; and (2) what languages can be defined by adding intersection or complement to the context free setting from Section 4.3.3 with the constant languages $\emptyset$, $\{\varepsilon\}$ and $\{a\}$ for every $a \in A$ as well as the operations of union, prefixing, concatenation and Kleene star.

Future Work: Part II

In Part II we see several possibilities for future work. Many of them are generalizations which appear due to the abstract category theoretic framework.

In Chapter 7, it is an open question whether the coincidence of Kleisli-CIAs and λ-CIAs for the maybe and powerset monads stated in Theorems 7.47 and 7.50 can be extended to non-polynomial functors; in particular taking analytic functors may be the next step. It would also be desirable to have a completely uniform proof of these two theorems (the proof already is uniform in part). A characterization of Kleisli-CIAs and λ-CIAs for the subdistribution monad may be possible by similar means as for the maybe and powerset monads; a characterization for the nonempty powerset monad remains an open issue.

We have seen how the important class of analytic functors yields results previously known only for finitary polynomial functors, for example in Theorem 6.19. It is an urgent open question whether the results of Chapter 8, in particular Definition 8.53 and Theorem 8.60 for NNRPS's, can be extended to analytic functors or even to weak pullback preserving functors; a starting point is given in Example 6.23(1).

An open topic of great relevance is to develop a theory of "complete Elgot monads". As explained in Remark 8.55(4), this is a missing part in the published literature which would allow for infinite terms in RPS's with effects.

Since we proved $J[\tau, \eta]^{-1}$ to be a weakly final coalgebra for the functor $\bar{\mathcal{H}}$ in Theorem 8.33, the question arises whether there is a final coalgebra and what it is. And clearly Chapter 8 leaves the question for a category theoretic semantics of interpreted RPS's with effects open for future research. Since this will involve algebras with effects, perhaps Kleisli-CIAs or λ-CIAs can be used which would strengthen the connection between systems of equations with effects and RPS's with effects.

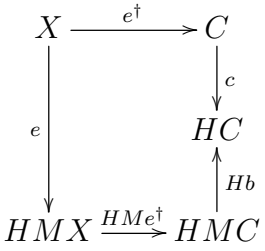
Reviewing Part II as a whole, one may ask for effects other than the ones corresponding to the five monads from Example 6.1. In particular, in Moggi's work [Mog91] there can be found computational effects like exceptions, side-effects, continuations, input or output. However, our framework does not seem to fit for the analysis of RPS's with these effects. It would be interesting to explore whether the approach via Lawvere theories instead of monads would allow for the treatment of different effects as the work of Abou-Saleh and Pattinson [ASP11] seems to indicate. Finally, in addition to our chess board example from the introduction to Part II and the examples illustrating particular definitions and results of Part II, it would be nice to find further applications of recursive program schemes with effects. For such applications our work could be used to find a practical definition format of effectful functions which can be implemented in programming languages or software tools for systems modeling.

Appendix A

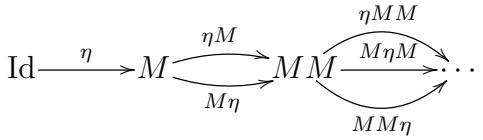
Proof of Theorem 4.64

In this appendix, we give the proof of Theorem 4.64 from page 88. It is based on Bartels' work [Bar04] and was extended to its present form (working for arbitrary cocomplete categories) by Milius. For the convenience of the reader, we first repeat the statement of this Theorem from Section 4.4.

Theorem A.1 (cf. [Bar04])**.** *Let $\lambda : MH \to HM$ be a distributive law of the pointed functor M over the functor H, and let $b : MC \to C$ be its λ-interpretation. Then for every* λ-equation $e : X \to HMX$ *there exists a unique solution, i. e., a unique morphism $e^\dagger : X \to C$ such that the diagram below commutes:*



Before we proceed to the proof of the statement we need some auxiliary constructions and lemmas. We begin by defining an endofunctor S on our cocomplete category as a colimit. We denote by M^n, $n \in \mathbb{N}$, the n-fold composition of M with itself. Now we consider the diagram D in the category of endofunctors on our base category given by the natural transformations in the picture below:



More formally, the diagram D is formed by all natural transformations

$$M^{i+j} \xrightarrow{M^i \eta M^j} M^{i+1+j} \qquad i, j \in \mathbb{N}\,.$$

Let S be a colimit of this diagram D:

$$S = \operatorname{colim} D \qquad \text{with injections } \mathsf{inj}^i : M^i \to S.$$

Then S is a pointed endofunctor with the point $\mathsf{inj}^0 : \mathrm{Id} = M^0 \to S$.

Recall that colimits in the category of endofunctors on our base category are formed objectwise. So for any object X, SX is a colimit of the diagram D at that object X with colimit injections $\mathsf{inj}^n_X : M^n X \to SX$, $n \in \mathbb{N}$. This implies that, for any endofunctor G the functor SG is a colimit with injections $\mathsf{inj}^n G : M^n G \to SG$.

The above definition of S appears in Bartels [Bar04]. Next we define additional data using the universal property of the colimits SM and SH:

1. a natural transformation $\chi : SM \to S$ uniquely determined by the commutativity of the triangles below:

$$\begin{array}{ccc} M^{n+1} & & \\ {\scriptstyle \mathsf{inj}^n M}\downarrow & \searrow^{\mathsf{inj}^{n+1}} & \\ SM & \xrightarrow[\chi]{} & S \end{array} \qquad \text{for all } n \in \mathbb{N}.$$

2. a natural transformation $\varepsilon : SM \to MS$ uniquely determined by the commutativity of the triangles below:

$$\begin{array}{ccc} M^{n+1} & & \\ {\scriptstyle \mathsf{inj}^n M}\downarrow & \searrow^{M\mathsf{inj}^n} & \\ SM & \xrightarrow[\varepsilon]{} & MS \end{array} \qquad \text{for all } n \in \mathbb{N}.$$

3. a natural transformation $\lambda^* : SH \to HS$; indeed, define first $\lambda^n : M^n H \to HM^n$ recursively as follows:

$$\lambda^0 = \mathrm{id}_H : H \to H;$$

$$\lambda^{n+1} = M^{n+1}H = MM^nH \xrightarrow{M\lambda^n} MHM^n \xrightarrow{\lambda M^n} HMM^n = HM^{n+1}\,.$$

Then λ^* is uniquely determined by the commutativity of the squares below:

$$\begin{array}{ccc} M^n H & \xrightarrow{\lambda^n} & HM^n \\ {\scriptstyle \mathsf{inj}^n H}\downarrow & & \downarrow{\scriptstyle H\mathsf{inj}^n} \\ SH & \xrightarrow[\lambda^*]{} & HS \end{array} \qquad \text{for all } n \in \mathbb{N}.$$

Observe that λ^* is a distributive law of the pointed endofunctor S over H; the unit law is the above square for the case $n = 0$.

We now need to verify that the three natural transformations above are well-defined. More precisely, we need to prove that those natural transformations are induced by appropriate cocones. For $\chi : SM \to S$ and $\lambda^* : SH \to HS$, this follows from Lemma 4.3.2 in Bartels' thesis [Bar04]. Hence, we make the explicit verification only for ε and leave the details for the other two natural transformations for the reader. To verify that the natural transformations $M\mathsf{inj}^n : M^{n+1} \to MS$ form a cocone for the appropriate diagram with colimit SM consider the triangles below:

$$\begin{array}{ccc} M^{1+i+j} & \xrightarrow{MM^i\eta M^j} & M^{1+i+1+j} \\ & {\scriptstyle M\mathsf{inj}^n}\searrow \quad \swarrow{\scriptstyle M\mathsf{inj}^{n+1}} & \\ & MS & \end{array} \qquad \text{for all } n \in \mathbb{N},\ n = i + j.$$

These triangles commute since $\mathsf{inj}^n : M^n \to S$ form a cocone.

Next, notice that in the definition of λ^* above there are two possible canonical choices for λ^{n+1}. We now show that these two choices are equal:

Lemma A.2. *For all natural numbers n we have the commutative square below:*

$$\begin{array}{ccc} M^{n+1}H & \xrightarrow{M^n\lambda} & M^nHM \\ {\scriptstyle M\lambda^n}\downarrow & & \downarrow{\scriptstyle \lambda^nM} \\ MHM^n & \xrightarrow[\lambda M^n]{} & HM^{n+1}\,. \end{array}$$

Proof. We prove the result by induction on n. The base case $n = 0$ is clear: both composites in the desired square are simply $\lambda : MH \to HM$. For the induction step we need to verify that the diagram below commutes:

$$\begin{array}{ccccc} & M^{n+1}MH & \xrightarrow{M^{n+1}\lambda = MM^n\lambda} & M^{n+1}HM & \\ & {\scriptstyle MM\lambda^n}\downarrow & & \downarrow{\scriptstyle M\lambda^nM} & \\ {\scriptstyle M\lambda^{n+1}} & MMHM^n & \xrightarrow{M\lambda M^n} & MHM^nM & {\scriptstyle \lambda^{n+1}M} \\ & {\scriptstyle M\lambda M^n}\downarrow & & \downarrow{\scriptstyle \lambda M^nM} & \\ & MHM^{n+1} & \xrightarrow[\lambda M^{n+1}]{} & HM^{n+1}M & \end{array}$$

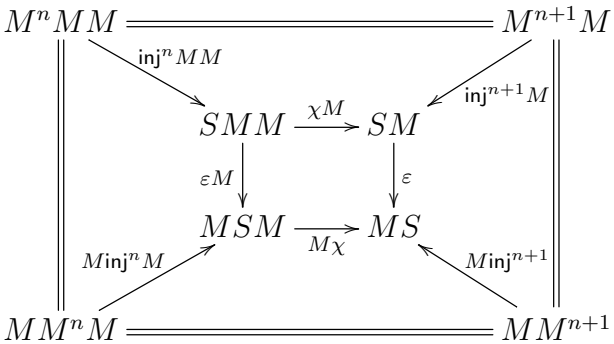
The left-hand and right-hand parts both commute due to the definition of λ^{n+1}. The lower square obviously commutes, and for the commutativity of the upper one apply the functor M to the induction hypothesis. Thus the desired outside square commutes. □

Next we need to establish a couple of properties connecting the three natural transformations χ, ε and λ^*.

Lemma A.3. *The following diagram of natural transformations commutes:*

$$\begin{array}{ccc} SMM & \xrightarrow{\chi M} & SM \\ {\scriptstyle \varepsilon M}\downarrow & & \downarrow{\scriptstyle \varepsilon} \\ MSM & \xrightarrow[M\chi]{} & MS\,. \end{array}$$

Proof. To verify that the square in the statement commutes we extend that square by the injections into the colimit SMM. This yields the following diagram:



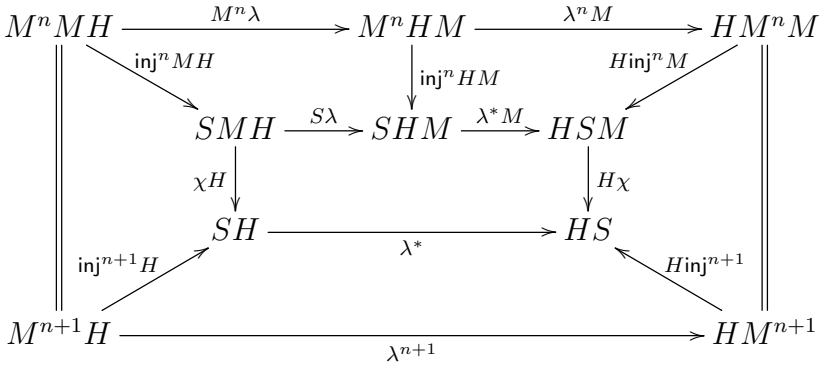
The left-hand and right-hand inner squares commute by the definition of ε, and the upper and lower inner square commute by the definition of χ. Since the outside commutes obviously, so does the desired middle square when extended by any injection inj^nMM of the colimit SMM. Thus, the desired middle square commutes. □

Lemma A.4. *The following square of natural transformations commutes:*

$$\begin{array}{ccccc} SMH & \xrightarrow{S\lambda} & SHM & \xrightarrow{\lambda^*M} & HSM \\ {\scriptstyle \chi H}\downarrow & & & & \downarrow{\scriptstyle H\chi} \\ SH & & \xrightarrow[\lambda^*]{} & & HS\,. \end{array}$$

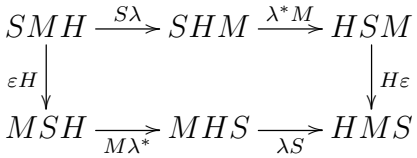
Proof. It suffices to verify that the desired square commutes when we extend it by an arbitrary colimit injection inj^nMH of SMH. To this end we consider

the diagram below:

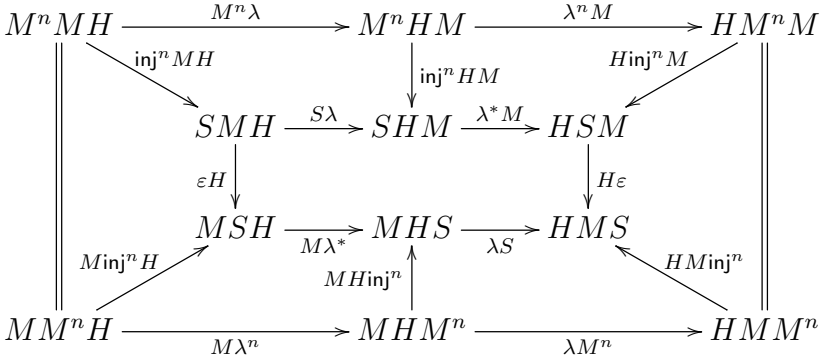


The left-hand and right-hand parts commute by the definition of χ, and the lower and the upper right-hand parts commute by the definition of λ^*. The upper left-hand part commutes by the naturality of inj^n. Finally, the outside commutes by the definition of λ^{n+1} together with Lemma A.2. Thus, the desired middle square commutes when extended by any colimit injection inj^nMH of the colimit SMH. □

Lemma A.5. *The following diagram of natural transformations commutes:*



Proof. Once more it is sufficient to verify that the desired square commutes when extended by any injection of the colimit SMH. So consider the diagram below:



The left-hand and right-hand parts commute by the definition of ε, and the lower left-hand and upper right-hand parts commute by the definition of λ^*.

The upper left-hand and the lower right-hand parts both commute due to the naturality of inj^n and λ, respectively. Finally, the outside commutes by Lemma A.2. Thus, the desired inner square commutes when extended by any colimit injection $\mathsf{inj}^n MH : M^n MH \to SMH$. □

We are now prepared to prove the statement of Theorem 4.64 (= Theorem A.1).

Proof of Theorem 4.64. Let $e : X \to HMX$ be any λ-equation. We form the following H-coalgebra:

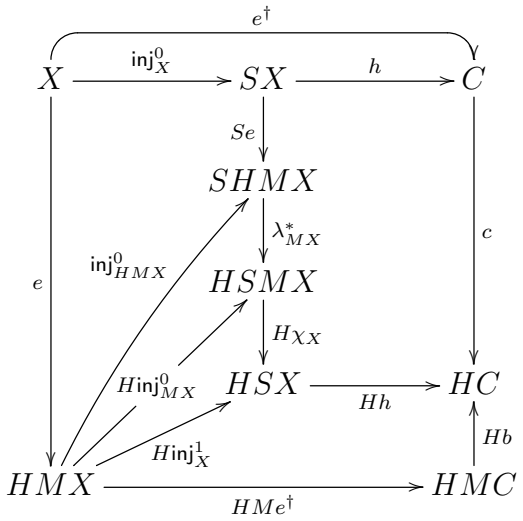
$$\bar{e} = SX \xrightarrow{Se} SHMX \xrightarrow{\lambda^*_{MX}} HSMX \xrightarrow{H\chi_X} HSX\,. \tag{A.1}$$

Since $c : C \to HC$ is a final H-coalgebra there exists a unique H-coalgebra homomorphism h from $(SX, \bar{e})$ to (C, c). We shall prove that the morphism

$$e^\dagger = X \xrightarrow{\mathsf{inj}^0_X} SX \xrightarrow{h} C \tag{A.2}$$

is the desired unique solution of the λ-equation e.

(1) $e^\dagger$ is a solution of e. It is our task to establish that the outside of the diagram below commutes (cf. the diagram in the statement of the theorem):

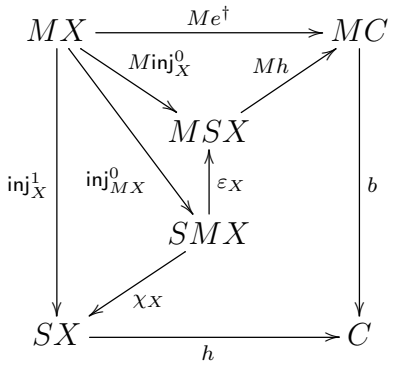


The upper part commutes by the definition of $e^\dagger$, and the upper right-hand square commutes since h is a coalgebra homomorphism. The upper left-hand part commutes due to the naturality of inj^0, the triangle below that commutes by the definition of λ^*, and the lowest triangle commutes by the definition of

χ. It remains to verify that the lowest part commutes. To this end we will now establish the following equation

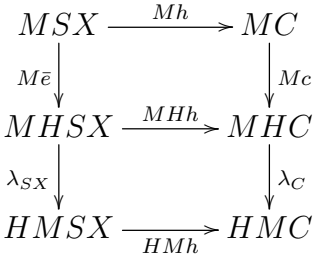
$$b \cdot Me^\dagger = h \cdot \mathsf{inj}^1_X \,. \tag{A.3}$$

Consider the diagram below:



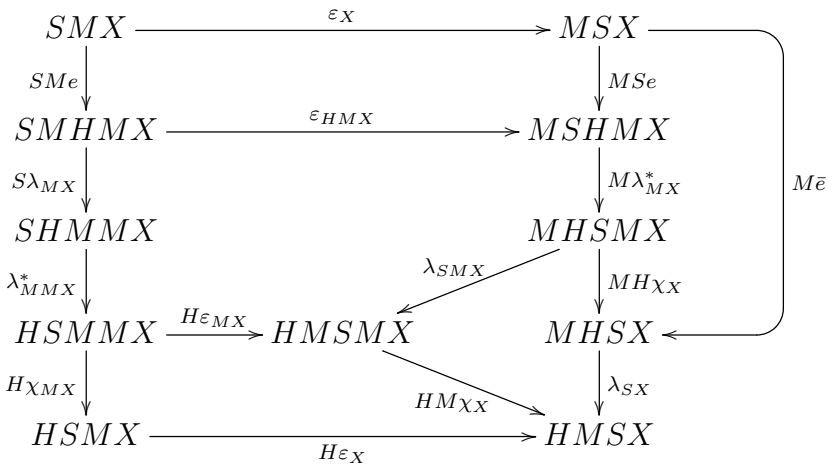
The upper triangle commutes by the definition of $e^\dagger$, the left-hand triangle commutes by the definition of χ and the inner triangle commutes by the definition of ε. In order to establish that the right-hand part commutes we will use that C is a final H-coalgebra. Thus, we shall exhibit H-coalgebra structures on the five objects and then show that all edges of the right-hand part of the diagram are H-coalgebra homomorphisms. Then by the uniqueness of coalgebra homomorphisms into the final coalgebra (C, c), we conclude that the desired part of the above diagram commutes.

For C, we use $c : C \to HC$, and for MC we use $\lambda_C \cdot Mc$. We already know that $b : MC \to C$ is a coalgebra homomorphism (see (4.14)). For SX, we use $\bar{e}$ from (A.1); again, we already know that $h : SX \to C$ is a coalgebra homomorphism. For MSX we use $\lambda_{SX} \cdot M\bar{e}$. The verification that Mh is a coalgebra morphism comes from the diagram below:



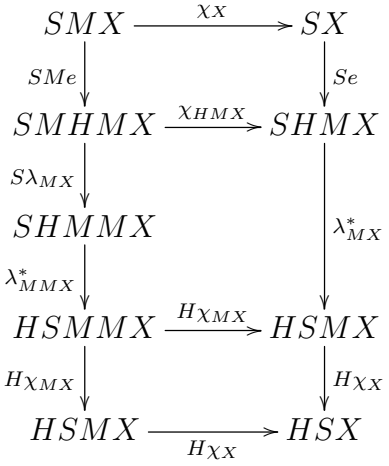
To see that the upper square commutes, remove M and recall that h is a coalgebra homomorphism from $(SX, \bar{e})$ to (C, c). The lower square commutes by the naturality of λ.

Now we show that $\varepsilon_X : SMX \to MSX$ is a coalgebra homomorphism, where the structure on SMX is the composite on the left below:



The upper square commutes by the naturality of ε, and the inner triangle commutes by the naturality of λ. To see that the right-hand part commutes, remove M and consider the definition of $\bar{e}$. The lowest part commutes due to Lemma A.3, and the middle part commutes by Lemma A.5.

Finally, we show that $\chi_X : SMX \to SX$ is a coalgebra homomorphism. To do this we consider the following diagram:



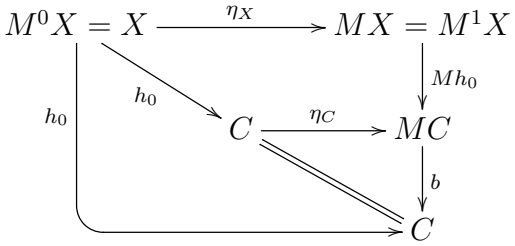
The upper square commutes by the naturality of χ, the middle square commutes by Lemma A.4, and the lower square commutes obviously. This concludes the proof that $e^\dagger$ is a solution of e.

(2) $e^\dagger$ in (A.2) is the unique solution of e. Suppose now that $e^\dagger$ is any solution of the λ-equation e. Recall that the object SX is a colimit of the diagram D at object X with the colimit injections $\mathsf{inj}^n_X : M^nX \to SX$, $n \in \mathbb{N}$.

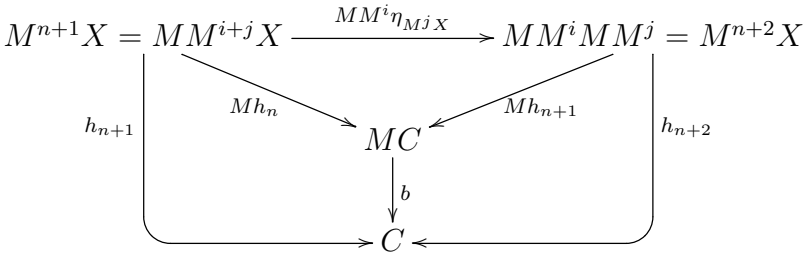
We will use the universal property of that colimit to define a morphism $h : SX \to C$. To this end we need to give a cocone $h_n : M^n X \to C$, $n \in \mathbb{N}$, for the appropriate diagram. We define this cocone inductively as follows:

$$h_0 = e^\dagger : M^0 X = X \to C;$$
$$h_{n+1} = M^{n+1}X = MM^n X \xrightarrow{Mh_n} MC \xrightarrow{b} C, \qquad n \in \mathbb{N}.$$

We now verify by induction on n that the morphisms h_n, $n \in \mathbb{N}$ do indeed form a cocone. For the base case consider the diagram below:

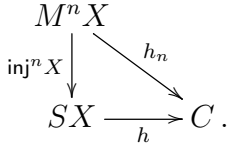


The upper part commutes by the naturality of η, the lower triangle commutes since $b : MC \to C$ is an algebra for the pointed endofunctor M, and the left-hand part is trivial. For the induction step consider for any natural number $n = i + j$ the following diagram:



This diagram commutes: for the upper triangle remove M and use the induction hypothesis, and the remaining two inner parts commute by the definition of h_{n+1} and h_{n+2}, respectively.

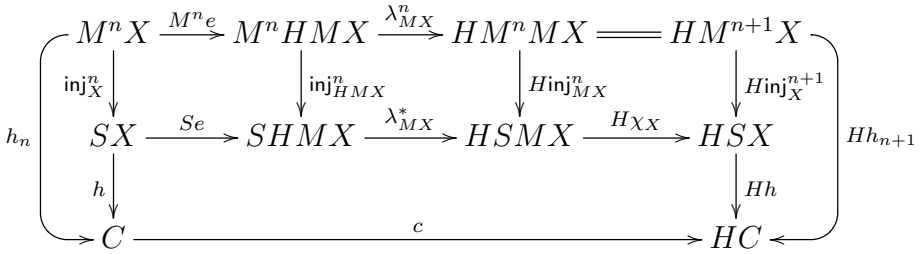
Now we obtain a unique morphism $h : SX \to C$ such that for any natural number n the triangle below commutes:



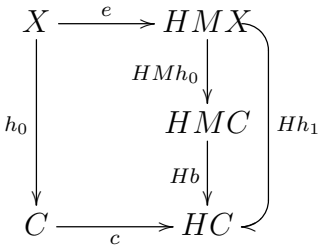
(A.4)

Next we show that $h : SX \to C$ is a coalgebra homomorphism from $(SX, \bar{e})$ to the final coalgebra (C, c). To this end we will now verify that the lower

part in the diagram below commutes:

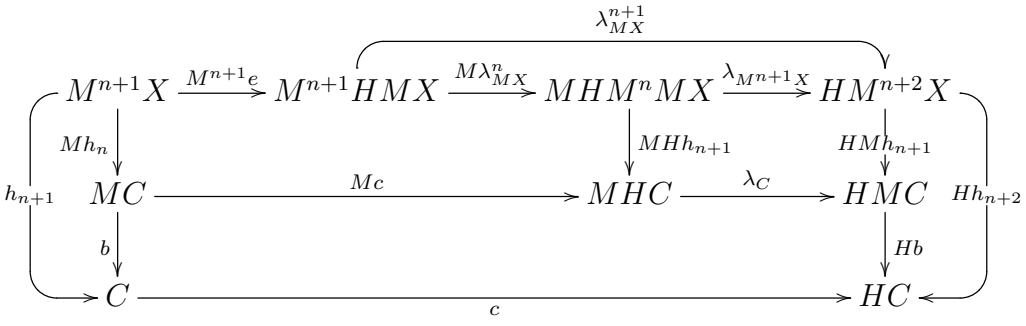


It suffices to show that the desired lower part commutes when extended by any colimit injection inj^n_X. Indeed, the left-hand part of the above diagram commutes by diagram (A.4), and for the commutativity of the right-hand part, remove H and use diagram (A.4) again. The upper left-hand square commutes by the naturality of inj^n, the upper middle square commutes by the definition of λ^*, and for the commutativity of the upper right-hand part remove H and use the definition of χ. It remains to verify that the outside of the diagram commutes. We will now prove this by induction on n. For the base case $n = 0$ we obtain the following diagram



This diagram commutes: for the commutativity of the right-hand part remove H and use the definition of h_1, and the left-hand part commutes since $h_0 = e^\dagger$ is a solution of the λ-equation e.

Finally, for the induction step we consider the diagram below:



We see that this diagram commutes as follows: the lower part commutes by the definition of $b : MC \to C$ (see (4.14)), the left-hand part commutes

by the definition of h_{n+1}, and for the commutativity of the right-hand part remove H and use the definition of h_{n+2}. The small upper part commutes by the definition of λ^{n+1}, the upper right-hand square commutes by the naturality of λ, and finally, to see the commutativity of the upper left-hand square remove M and use the induction hypothesis.

We have finished the proof that $h : SX \to C$ is a coalgebra homomorphism from $(SX, \bar{e})$ to the final coalgebra (C, c). Since h is uniquely determined, it follows that the solution $e^\dagger = h \cdot \mathsf{inj}_X^0$ is uniquely determined, too. This completes our proof. □

Appendix B

Tables

B.1 List of Symbols

Sets and Elements	
$\mathbb{N}$	set of natural numbers including 0
$\mathbb{N}_\bot$	$\mathbb{N} \cup \{\bot\}$
$\mathbb{R}$	set of real numbers
$\mathbb{R}^\omega$	set of streams
$\mathbb{R}^\omega_{\text{ep}}$	set of eventually periodic streams
$\mathbb{R}^\omega_{\text{ra}}$	set of rational streams
C_{ra}	set of rational strongly extensional finitely branching trees
C_{reg}	set of regular languages
C_{cf}	set of context-free languages
ε	empty word (in Part I)

Orders	
$\leq, \sqsubseteq$	(complete) partial orders
$\bigvee$	join of an ω-chain
$\bigwedge$	meet of an ω-chain
$\bot$	least element of some (complete) partial order
$\top$	greatest element of some (complete) partial order

Signatures and Operation Symbols

Σ, Γ	signatures
Σ_n	set of n-ary operation symbols from Σ
σ, τ	operation symbols from a signature (in Chapter 2 and Part II)
ϕ, ψ, χ, ρ	newly defined operations in a RPS (in Part II)

Categories

$\mathcal{C}$	category
$[\mathcal{B}, \mathcal{C}]$	category of functors $\mathcal{B} \to \mathcal{C}$ and natural transformations
$\mathcal{C}_M$	Kleisli category of $M : \mathcal{C} \to \mathcal{C}$
$(\mathcal{C}, I, \otimes, \mathsf{ul}, \mathsf{ur}, \mathsf{a}, \mathsf{c})$	symmetric monoidal category
Set	category of sets and functions
Vec	category of real vector spaces and linear transformations
Class	category of classes and functions
H-Alg	category of H-algebras and homomorphisms
H-CIA	category of completely iterative H-algebras
$\bar{H}$-CIA	category of Kleisli-CIAs (CIAs for $\bar{H}$)
H-CIA_λ	category of λ-CIAs for H
H-CEA	category of complete Elgot algebras for H
(M, η, μ)-Alg	category of Eilenberg-Moore algebras for (M, η, μ)
H-Coalg	category of H-coalgebras and homomorphisms
(D, ε)-Coalg	category of coalgebras for the copointed functor (D, ε)
NatB	category of natural numbers and bijections

Morphisms

outl, outr	left/right product projections
$\langle f, g \rangle$	pairing of morphisms f, g
inl, inr	left/right coproduct injections
$[f, g]$	copairing of morphisms f, g
can	canonical arrow $[G\mathsf{inl}, G\mathsf{inr}] : GX + GY \to G(X + Y)$
inj_i	i-th injection
π_i	i-th projection
i	initial arrow
$!$	final arrow

Functors

H_Σ	polynomial functor for the signature Σ
G, G', H, H', K	functors
Id	identity functor
C_X	constant functor at the object X
$\bar{H}$	lifting of H to $\mathcal{C}_M$
J	inclusion functor $\mathcal{C} \to \mathcal{C}_M$
V	canonical right adjoint $\mathcal{C}_M \to \mathcal{C}$
$\mathcal{H}$	functor $H(-) + \mathrm{Id}$ on $[\mathcal{C}, \mathcal{C}]$
$\bar{\mathcal{H}}$	lifting of $\mathcal{H}$ to $[\mathcal{C}, \mathcal{C}]_{\mathcal{M}}$
$\mathcal{P}_\mathrm{f}$	finite powerset functor
$\mathcal{P}_\mathrm{c}$	countable powerset functor
$\mathcal{P}_\kappa$	κ-ary powerset functor
$\mathcal{P}$	powerset functor
$\mathcal{E}$	embedding $\mathsf{NatB} \to \mathsf{Set}$
P	polynomial functor underlying an analytic functor

(Co-)Pointed Functors and Monads

(M, η^M)	pointed functor
$(M, \eta^M, \mu^M), (N, \eta^N, \mu^N)$	monads
$(M, \eta^M, \mu^M, \bar{M}, \bar{\mu}^M, \vartheta^M)$	idealized monad
$(F^H, \eta^H, \mu^H), \kappa^H$	free monad on H with universal natural transformation
$(T^H, \eta^H, \mu^H), \kappa^H$	free completely iterative monad on H with universal natural transformation
$\mathcal{M}$	monad $(M \cdot -, \eta^M \cdot -, \mu^M \cdot -)$ on $[\mathcal{C}, \mathcal{C}]$
$(\mathcal{P}, \eta, \mu)$	powerset monad
$(\mathcal{P}^+, \eta^+, \mu^+)$	nonempty powerset monad
$(\mathcal{D}, \eta, \mu)$	subdistribution monad
(D, ε^D)	copointed functor
$(Q^H, \varepsilon^H), \iota^H$	cofree copointed functor on H with (co)universal natural transformation

Natural Transformations and Distributive Laws

α	natural transformation
$\alpha * \beta$	parallel composition of natural transformations
$(\theta, \bar{\theta})$	idealized monad morphism
$\alpha^{\#}$	unique monad morphism induced by $\alpha : H \to M$
$\widehat{\mathsf{inl}}$, $\widehat{\mathsf{inr}}$	liftings of $\mathsf{inl}/\mathsf{inr}$ to monad morphisms between free monads
ε	counit of an adjunction (in Part II)
ϵ	quotienting transformation $P \to H$
ℓ	natural transformation inducing a distributive law
λ	distributive law
λ'	extension of a distributive law $\lambda : HM \to MH$ to a distributive law of monads
δ	distributive law of monads
Λ	distributive law $\mathcal{HM} \to \mathcal{MH}$ induced by $\lambda : HM \to MH$

Algebras and Coalgebras

(A, a)	algebra
(I, i)	initial algebra
(C, b)	algebra induced by an abstract GSOS rule ℓ (ℓ-interpretation) or by a distributive law λ (λ-interpretation)
$(A, \widehat{a})$	Eilenberg-Moore algebra for a free monad induced by (A, a)
$(A, \widetilde{a})$	Eilenberg-Moore algebra for a free CIM induced by (A, a)
(S, s)	coalgebra
(C, c)	final coalgebra
(V, c)	final coalgebra of $\mathcal{P} : \mathsf{Class} \to \mathsf{Class}$
ϕ_X^H	free H-algebra on X
τ_X^H	final $(H + X)$-coalgebra

Equation Morphisms, RPS's and Solutions

e	(flat) equation morphism, RPS
$h \bullet e$	(flat) equation morphism with parameters renamed by h
$f \blacksquare e$	composite of (flat) equation morphisms e and f
e'	morphism guarding e
$\bar{e}$	equation morphism over e
$e^{\dagger}$	solution, uninterpreted solution (RPS)
$e^{\ddagger}$	interpreted solution (RPS)

Streams

σ, τ	streams (in Part I)
ones	stream of ones
twos	stream of twos
even	stream of even natural numbers
even$^+$	stream of even natural numbers without 0
odd	stream of odd natural numbers
squares	stream of squares of natural numbers

B.2 List of Abbreviations

CCS	calculus of communicating systems
CEA	complete Elgot algebra
CIA	completely iterative algebra
CIM	completely iterative monad
CNF	conjunctive normal form
CPO	complete partial order
GNF	Greibach normal form
GSOS	guarded structured operation semantics
ℓ-RPS	recursive program scheme w. r. t. ℓ
λ-CIA	completely λ-iterative algebra
M-RPS	recursive program scheme with M-effects
NNRPS	nonempty nondeterministic recursive program scheme
RPS	recursive program scheme
SOS	structured operational semantics

Index

Bibliography

[AAMV03] Peter Aczel, Jiří Adámek, Stefan Milius, and Jiří Velebil. Infinite trees and completely iterative theories: A coalgebraic view. *Theoret. Comput. Sci.*, 300:1–45, 2003.

[Ace94] Luca Aceto. GSOS and finite labelled transition systems. *Theoret. Comput. Sci.*, 131(1):181–195, 1994.

[Acz88] Peter Aczel. *Non-Well-Founded Sets*, volume 14 of *CLSI Lecture Notes*. CLSI Publications, Stanford, 1988.

[Adá05] Jiří Adámek. Introduction to coalgebra. *Theory Appl. Categ.*, 14:157–199, 2005.

[AFV01] Luca Aceto, Wan J. Fokking, and Chris Verhoef. Structural operational semantics. In J. A. Bergstra, A. Ponse, and S. A. Smolka, editors, *Handbook of Process Algebra*, pages 197–292. Elsevier, 2001.

[AHS09] Jiří Adámek, Horst Herrlich, and George E. Strecker. *Abstract and Concrete Categories—The Joy of Cats*. Dover Publications Inc., 2009.

[AM06] Jiří Adámek and Stefan Milius. Terminal coalgebras and free iterative theories. *Inform. and Comput.*, 204:1139–1172, 2006.

[AMV03] Jiří Adámek, Stefan Milius, and Jiří Velebil. Free iterative theories: A coalgebraic view. *Math. Structures Comput. Sci.*, 13:259–320, 2003.

[AMV04] Jiří Adámek, Stefan Milius, and Jiří Velebil. On coalgebras based on classes. *Theoret. Comput. Sci.*, 316:3–23, 2004.

[AMV06a] Jiří Adámek, Stefan Milius, and Jiří Velebil. Elgot algebras. *Log. Methods Comput. Sci.*, 2(5:4):31 pp., 2006.

[AMV06b] Jiří Adámek, Stefan Milius, and Jiří Velebil. Iterative algebras at work. *Math. Structures Comput. Sci.*, 16:1085–1131, 2006.

[AMV11] Jiří Adámek, Stefan Milius, and Jiří Velebil. Elgot theories: A new perspective of the equational properties of iteration. *Math. Structures Comput. Sci.*, 21(2):417–480, 2011.

[AN80] André Arnold and Maurice Nivat. Formal computations of non deterministic recursive program schemes. *Math. Systems Theory*, 13:219–236, 1980.

[AP04] Jiří Adámek and H.-E. Porst. On tree coalgebras and coalgebra presentations. *Theoret. Comput. Sci.*, 311(1–3):257–283, 2004.

[AR94] Jiří Adámek and Jiří Rosický. *Locally Presentable and Accessible Categories*, volume 189 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, 1994.

[ASP11] Faris Abou-Saleh and Dirk Pattinson. Towards effects in mathematical operational semantics. *Electron. Notes Theor. Comput. Sci.*, 276:81–104, 2011.

[AT90] Jiří Adámek and Věra Trnková. *Automata and Algebras in Categories*, volume 37 of *Mathematics and its Applications*. Kluwer Academic Publishers, 1990.

[Bar70] Michael Barr. Coequalizers and free triples. *Math. Z.*, 116:307–322, 1970.

[Bar03] Falk Bartels. Generalized coinduction. *Math. Structures Comput. Sci.*, 13(2):321–348, 2003.

[Bar04] Falk Bartels. *On Generalized Coinduction and Probabilistic Specification Formats*. PhD thesis, Vrije Universiteit Amsterdam, 2004.

[Bec69] Jon Beck. Distributive laws. In *Seminar on Triples and Categorical Homology Theory*, volume 80 of *Lecture Notes in Math.*, pages 119–140. Springer Berlin Heidelberg, 1969.

[BIM95] Bard Bloom, Sorin Istrail, and Albert R. Meyer. Bisimulation can't be traced. *J. ACM*, 42(1):232–268, 1995.

[BM96] Jon Barwise and Lawrence S. Moss. *Vicious circles*. CLSI publications, Stanford, 1996.

[BMR12] Marcello M. Bonsangue, Stefan Milius, and Jurriaan Rot. On the specification of operations on the rational behavior of systems. In Bas Luttik and Michel A. Reniers, editors, *Proc. Combined Workshop on Expressiveness in Concurrency and Structural Operational Semantics (EXPRESS/SOS'12)*, volume 89 of *Electron. Proc. Theoret. Comput. Sci.*, pages 3–18. Open Publishing Association, 2012.

[Bou80] Gérard Boudol. Sémantique opérationelle et algébrique des programmes récursifs non déterministes. These d'Etat, Université de Paris VII, 1980.

[CKW90] Aurelio Carboni, G. Max Kelly, and Richard J. Wood. A 2-categorical approach to change of base and geometric morphisms I. Technical Report 90-1, Department of Pure Mathematics, University of Sydney, 1990.

[Cou83] Bruno Courcelle. Fundamental properties of infinite trees. *Theoret. Comput. Sci.*, 25(2):95–169, 1983.

[CUV06] Venanzio Capretta, Tarmo Uustalu, and Varmo Vene. Recursive coalgebras from comonads. *Inform. and Comput.*, 204:437–468, 2006.

[EBT78] Calvin C. Elgot, Stephen L. Bloom, and Ralph Tindell. On the algebraic structure of rooted trees. *J. Comput. System Sci.*, 16:361–399, 1978.

[FT01] Marcello Fiore and Danielle Turi. Semantics of name and value passing. In *Proc. 16th Annual Symposium on Logic in Computer Science (LICS'01)*, pages 93–104. IEEE Computer Society, 2001.

[GLM03] Neil Ghani, Christoph Lüth, and Federico De Marchi. Solving algebraic equations using coalgebra. *Theor. Inform. Appl.*, 37:301–314, 2003.

[Gre65] Sheila A. Greibach. A new normal-form theorem for context-free phrase structure grammars. *J. ACM*, 12(1):42–52, 1965.

[GS13] Sergey Goncharov and Lutz Schröder. A coinductive calculus for asynchronous side-effecting processes. *Inform. and Comput.*, 231:204–232, 2013.

[Gue81] Irène Guessarian. *Algebraic Semantics*, volume 99 of *Lecture Notes in Comput. Sci.* Springer, 1981.

[Gum99] Heinz Peter Gumm. Elements of the general theory of coalgebras. Course Notes for the First Southern African Summer School and Workshop on Logic, Universal Algebra, and Theoretical Computer Science (LUATCS '99), Johannesburg, 1999.

[HJ05] Ichiro Hasuo and Bart Jacobs. Context-free languages via coalgebraic trace semantics. In J. Fiadeiro et al., editor, *Proc. Algebra and Coalgebra in Computer Science: First International Conference (CALCO'05)*, volume 3629 of *Lecture Notes in Comput. Sci.*, pages 213–231. Springer, 2005.

[HJS07] Ichiro Hasuo, Bart P. F. Jacobs, and Ana Sokolova. Generic trace semantics via coinduction. *Log. Methods Comput. Sci.*, 3(4:11):36 pp., 2007.

[HMU07] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley Longman, Amsterdam, 3rd edition, 2007.

[HP07] Martin Hyland and John Power. The category theoretic understanding of universal algebra: Lawvere theories and monads. *Electron. Notes Theor. Comput. Sci.*, 172:437–458, 2007.

[Jac04] Bart P. F. Jacobs. Trace semantics for coalgebras. *Electron. Notes Theor. Comput. Sci.*, 106:167–184, 2004.

[Jac06a] Bart P. F. Jacobs. A bialgebraic review of deterministic automata, regular expressions and languages. In *Goguen Festschrift*, volume 4060 of *Lecture Notes in Comput. Sci.*, 2006.

[Jac06b] Bart P. F. Jacobs. Distributive laws for the coinductive solution of recursive equations. *Inform. and Comput.*, 204(4):561–587, 2006.

[JGH11] Mauro Jaskelioff, Neil Ghani, and Graham Hutton. Modularity and implementation of mathematical operational semantics. In Venanzio Capretta and Conor McBride, editors, *Proc. Second Workshop on Mathematically Structured Functional Programming (MSFP 2008)*, volume 229 of *Electron. Notes Theor. Comput. Sci.*, pages 75–95. Elsevier Science B.V., 2011.

[Joy81] André Joyal. Une théorie combinatoire des séries formelles. *Adv. Math.*, 42:1–82, 1981.

[Joy86] André Joyal. Foncteurs analytiques et espèces de structures. In G. Labelle and P. Leroux, editors, *Combinatoire énumérative (Lecture Notes in Math. 1234)*, pages 126–159. Springer, 1986.

[JR97] Bart P. F. Jacobs and Jan J. M. M. Rutten. A tutorial on (co)algebras and (co)induction. *Bulletin of EATCS*, 62:222–259, 1997.

[JS93] André Joyal and Ross Street. Braided tensor categories. *Adv. Math.*, 102:20–78, 1993.

[Kel80] G. Max Kelly. A unified treatment of transfinite constructions for free algebras, free monoids, colimits, associated sheaves, and so on. *Bull. Austral. Math. Soc.*, 22(1):1–83, 1980.

[Kle65] Heinrich Kleisli. Every standard construction is induced by a pair of adjoint functors. *Proc. Amer. Math. Soc.*, 16(3):544–546, 1965.

[Kli04] Bartek Klin. Adding recursive constructs to bialgebraic semantics. *J. Log. Algebr. Program.*, (60-61):259–286, 2004.

[Kli09] Bartek Klin. Structural operational semantics for weighted transition systems. In Jens Palsberg, editor, *Semantics and Algebraic Specification: Essays Dedicated to Peter D. Mosses on the Occasion of His 60th Birthday*, volume 5700 of *Lecture Notes in Comput. Sci.*, pages 121–139. Springer, 2009.

[Kli11] Bartek Klin. Bialgebras for structural operational semantics: an introduction. *Theoret. Comput. Sci.*, 462(38):5043–5069, 2011.

[Koc70] Anders Kock. Monads on symmetric monoidal closed categories. *Arch. Math. (Basel)*, 21:1–10, 1970.

[Koc72] Anders Kock. Strong functors and monoidal monads. *Arch. Math. (Basel)*, 23:113–120, 1972.

[KP04] Marco Kick and A. John Power. Modularity of behaviours for mathematical operational semantics. *Electron. Notes Theor. Comput. Sci.*, 106:185–200, 2004.

[KPS06] Marco Kick, A. John Power, and Alex Simpson. Coalgebraic semantics for timed processes. *Inform. and Comput.*, 204:588–609, 2006.

[Lam68] Joachim Lambek. A fixpoint theorem for complete categories. *Math. Z.*, 103(2):151–161, 1968.

[LPW00] Marina Lenisa, A. John Power, and Hiroshi Watanabe. Distributivity for endofunctors, pointed and co-pointed endofunctors, monads and comonads. In Horst Reichel, editor, *Proc. Coalgebraic Methods in Computer Science*, volume 33 of *Electron. Notes Theor. Comput. Sci.* Elsevier Amsterdam, 2000.

[LPW04] Marina Lenisa, A. John Power, and Hiroshi Watanabe. Category theory for operational semantics. *Theoret. Comput. Sci.*, 327:135–154, 2004.

[MA86] Ernest G. Manes and Michael A. Arbib. *Algebraic Approaches to Program Semantics*. Texts and Monographs in Computer Science. Springer, New York, 1986.

[Mac98] Saunders MacLane. *Categories for the Working Mathematician*. Graduate Texts in Mathematics. Springer, Berlin, 2nd edition, 1998. Edited by F. W. Gehring and P. R. Halmos.

[Mil89] Robin Milner. *Communication and Concurrency*. International Series in Computer Science. Prentice Hall, 1989.

[Mil05] Stefan Milius. Completely iterative algebras and completely iterative monads. *Inform. and Comput.*, 196:1–41, 2005.

[Mil10] Stefan Milius. A sound and complete calculus for finite stream circuits. In *Proc. 25th Annual Symposium on Logic in Computer Science (LICS'10)*, pages 449–458. IEEE Computer Society, 2010.

[MM06] Stefan Milius and Lawrence S. Moss. The category theoretic solution of recursive program schemes. *Theoret. Comput. Sci.*, 366:3–59, 2006. Fundamental study; a corrigendum appeared in *Theoret. Comput. Sci.* 403:409–415, 2008.

[MM09] Stefan Milius and Lawrence S. Moss. Equational properties of recursive program scheme solutions. *Cah. Topol. Gèom. Diffèr. Catèg.*, 50:23–66, 2009.

[MMS10] Stefan Milius, Lawrence S. Moss, and Daniel Schwencke. CIA structures and the semantics of recursion. In C. H. Luke Ong, editor, *Proc. Foundations of Software Science and Computation Structures (FoSSaCS)*, volume 6014 of *Lecture Notes in Comput. Sci.*, pages 312–327. Springer, 2010.

[MMS13] Stefan Milius, Lawrence S. Moss, and Daniel Schwencke. Abstract GSOS rules and a compositional treatment of recursive definitions. *Log. Methods Comput. Sci.*, 9(3:28):52 pp., 2013.

[Mog91] Eugenio Moggi. Notions of computation and monads. *Inform. and Comput.*, 93(1):55–92, 1991.

[Mos03] Lawrence S. Moss. Recursion and corecursion have the same equational logic. *Theoret. Comput. Sci.*, 294(1/2):233–267, 2003.

[MPS09] Stefan Milius, Thorsten Palm, and Daniel Schwencke. Complete iterativity for algebras with effects. In Alexander Kurz, Marina Lenisa, and Andrzej Tarlecki, editors, *Proc. Coalgebraic and Algebraic Methods in Computer Science (CALCO'09)*, volume 5728 of *Lecture Notes in Comput. Sci.*, pages 34–48. Springer, 2009.

[Mul94] Philip S. Mulry. Lifting theorems for kleisli categories. In S. Brookes, M. Main, A. Melton, M. Mislove, and D. Schmidt, editors, *Mathematical Foundations of Programming Semantics—9th international conference*, volume 802 of *Lecture Notes in Comput. Sci.*, pages 304–319. Springer Verlag, 1994.

[Niv75] Maurice Nivat. On the interpretation of recursive polyadic program schemes. In *Symposia Mathematica XV*, pages 255–281. Academic Press, New York, 1975.

[Plo81] Gordon D. Plotkin. A structural approach to operational semantics. Technical Report DAIMI FN-19, Computer Science Department, Aarhus University, 1981.

[Plo04] Gordon D. Plotkin. The origins of structural operational semantics. *J. Log. Algebr. Program.*, 60–61:3–15, 2004.

[Poi82] Axel Poigné. On effective computations of nondeterministic schemes. In *Symposium on Programming*, volume 137 of *Lecture Notes in Comput. Sci.*, pages 323–336. Springer, 1982.

[Pow03] A. John Power. Towards a theory of mathematical operational semantics. In H. Peter Gumm, editor, *Proc. Coalgebraic Methods in Computer Science (CMCS'03)*, volume 82 of *Electron. Notes Theor. Comput. Sci.*, pages 257–272. Elsevier Science B.V., 2003.

[Pow06] A. John Power. Countable lawvere theories and computational effects. In Anthony K. Seda, Ted Hurley, Michel Schellekens, Micheál Mac an Airchinnigh, and Glenn Strong, editors, *Proc. Third Irish Conference on the Mathematical Foundations of Computer Science and Information Technology (MFCSIT 2004)*, volume 161 of *Electron. Notes Theor. Comput. Sci.*, pages 59–71. Elsevier Science B.V., 2006.

[Rut00] Jan J. M. M. Rutten. Universal coalgebra: a theory of systems. *Theoret. Comput. Sci.*, 249(1):3–80, 2000.

[Rut05a] Jan J. M. M. Rutten. A coinductive calculus of streams. *Math. Structures Comput. Sci.*, 15(1):93–147, 2005.

[Rut05b] Jan J. M. M. Rutten. A tutorial on coinductive stream calculus and signal flow graphs. *Theoret. Comput. Sci.*, 343(3):443–481, 2005.

[Rut08] Jan J. M. M. Rutten. Rational streams coalgebraically. *Log. Methods Comput. Sci.*, 4(3:9):22 pp., 2008.

[Sch10a] Daniel Schwencke. Coequational logic for accessible functors. *Inform. and Comput.*, 208(12):1469–1489, 2010.

[Sch10b] Daniel Schwencke. Recursive program schemes with effects. In B. P. F. Jacobs, M. Niqui, J. J. M. M. Rutten, and A. Silva, editors, *CMCS '10 Short Contributions*, CWI Technical Report SEN–1004, pages 20–22, 2010.

[Sch11] Daniel Schwencke. A category theoretic view of nondeterministic recursive program schemes. In Marc Bezem, editor, *Computer Science Logic (CSL'11) - 25th International Workshop/20th Annual Conference of the EACSL*, volume 12 of *Leibnitz International Proceedings in Computer Science (LIPIcs)*, pages 496–511. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2011.

[Sim11] Harold Simmons. *An Introduction to Category Theory*. Cambridge University Press, 2011.

[SR07] Alexandra Silva and Jan J. M. M. Rutten. Behavioural differential equations and coinduction for binary trees. In D. Leivant and R. de Queiroz, editors, *Proc. WoLLIC 2007*, volume 4576 of *Lecture Notes in Comput. Sci.*, pages 322–336. Springer, 2007.

[SR10] Alexandra Silva and Jan J. M. M. Rutten. A coinductive calculus of binary trees. *Inform. and Comput.*, 208(5):578–593, 2010.

[TP97] Danielle Turi and Gordon D. Plotkin. Towards a mathematical operational semantics. In *Proc. Logic in Computer Science (LICS)*, 1997.

[Trn77] Věra Trnková. Relational automata in a category and theory of languages. In Marek Karpinski, editor, *Fundamentals of Computation Theory, Proceedings of the 1977 International FCT-Conference, Poznan-Kórnik*, volume 56 of *Lecture Notes in Comput. Sci.*, pages 340–355. Springer Berlin Heidelberg New York, 1977.

[Tur01] Daniele Turi. Category theory lecture notes. Available at `http://www.dcs.ed.ac.uk/home/dt/CT`, 2001.

[UVP01] Tarmo Uustalu, Varmo Vene, and Alberto Pardo. Recursion schemes from comonads. *Nordic J. Comput.*, 8(3):366–390, 2001.

[vO95] Jaap van Oosten. Basic category theory. BRICS Lecture Series LS-95-1, 1995. Revised version (2002) available at `http://www.staff.science.uu.nl/~ooste110/onderwijs.html`.

[Wad95] Philip Wadler. Monads for functional programming. In Johan Jeuring and Erik Meijer, editors, *Advanced Functional Programming*, volume 925 of *Lecture Notes in Comput. Sci.*, pages 24–52. Springer Berlin Heidelberg, 1995.

[WBR11] Joost Winter, Marcello M. Bonsangue, and Jan J. M. M. Rutten. Context-free languages, coalgebraically. In Andrea Corradini, Bartek Klin, and Corina Cîrstea, editors, *Proc. Algebra and Coalgebra in Computer Science (CALCO'11)*, volume 6859 of *Lecture Notes Comput. Sci.*, pages 359–376. Springer, 2011.

[Wor05] James Worrell. On the final sequence of a finitary set functor. *Theoret. Comput. Sci.*, 338:184–199, 2005.